ORIGINAL RESEARCH

Multiclass patent document classification

Chaitanya Anne, Avdesh Mishra, Md Tamjidul Hoque*, Shengru Tu

University of New Orleans, New Orleans, Louisiana, USA

Received: September 29, 2017 **Accepted:** December 1, 2017 **Online Published:** December 15, 2017
DOI: 10.5430/air.v7n1p1 **URL:** http://dx.doi.org/10.5430/air.v7n1p1

ABSTRACT

This article addresses patent document classification problem into fifteen different categories or classes, where some classes overlap with each other for practical reasons. For the development of the classification model using machine learning techniques, useful features have been extracted from the given documents. The features are used to classify patent document as well as to generate useful tag-words. The overall objective of this work is to systematize NASA's patent management, by developing a set of automated tools that can assist NASA to manage and market its portfolio of intellectual properties (IP), and to enable easier discovery of relevant IP by users. We have identified an array of methods that can be applied such as k-Nearest Neighbors (kNN), two variations of the Support Vector Machine (SVM) algorithms, and two tree based classification algorithms: Random Forest and J48. The major research steps in this paper consist of filtering techniques for variable selection, information gain and feature correlation analysis, and training and testing potential models using effective classifiers. Further, the obstacles associated with the imbalanced data were mitigated by adding pseudo-synthetic data wherever appropriate, which resulted in a superior SVM classifier based model.

Key Words: Machine learning algorithms, Patent classification, Imbalanced data, SVM, Multi-class classification

1. INTRODUCTION

Document classification is the process of classifying a document into a predefined category. It plays a vital role in managing large number of text documents by automating the document classification task. The two main approaches in machine learning for document classification are supervised learning,[1] where the model is trained with labeled documents, and unsupervised learning, where the predefined category or training data is not available for the classification task, rather clusters are made to observe the natural grouping of the documents. Further, the given document in supervised approach can be labeled as a single-class document as well as multi-class documents. Multi-class labeled document classification is relatively challenging compared to the single-class labeled document.[2] In addition to supervised and unsupervised learning there exist a special form of learning known as semi-supervised learning (SSL).[3] SSL falls in between supervised and unsupervised learning. In addition to labeled data, the algorithm makes use of unlabeled data for training. In SSL, the data set $X = (x_i)_{i\in[n]}$ can be divided into two parts: the points $X_l = (x_1, \cdots, x_l)$ for which labels $Y_l = (y_1, \cdots, y_l)$ are provided, and the points $X_u = (x_{l+1}, \cdots, x_{l+u})$, the labels of which are unknown. In practice, it is found that SSL is most useful whenever there are far more unlabeled data than labeled. This is followed by the fact that obtaining labels is time, labor and expense intensive. SSL methods use unlabeled data to either modify or reprioritize hypothesis obtained from labeled data alone.

Supervised learning models are widely applicable because often the possible categories are predetermined based on

*Correspondence: Md Tamjidul Hoque; Email: thoque@uno.edu; Address: Computer Science, University of New Orleans, 2000 Lakeshore Dr, New Orleans, LA 70148, USA.

business activities. Document classification mainly consists of document representation (vector form), feature selection, feature extraction, application of machine learning algorithm on the data for evaluation. In this paper we performed classification task on NASA's patent documents, available in html format and downloaded from the U.S. Patent Office.[4] We applied five machine learning methods k-Nearest Neighbors (kNN), two variations of Support Vector Machine (SVM) (with RBF and PolyKernel), and two tree-based classifier (Random Forest and J48), to carry out the initial classification on NASA patent data. We have assessed the accuracies of the initial classification with 5 fold cross-validation (CV). Based on the results, we conclude that the SVM performed well on the available patent data compared to the other algorithms.

To make document search convenient to the users, tag-words can be generated from the training files as well as from the input files based on the information gain[5] of the useful word or term calculated from the given document. The generation of these tag-words will provide further fine-grained details about patents combining the subcategories information, which can assist in search activities and document management.

To perform the entire document classification and management task, we have developed the proposed tool in Java, available online (`https://github.com/tamjidul/Patent_Classifier`). This tool uses the training data samples that are available in the study,[4] builds model and then predicts the category of the input document. Along with the classification, probability distribution of the document could be viewed in pie chart by utilizing the probability values which are generated by the classifier. This helps user predict multi-labeled document and enable to place the document in the most appropriate category.

One of the challenges in this paper was the existence of the imbalanced dataset. Imbalance in training dataset implies significant difference in the number of samples present among the classes. Imbalance dataset is problematic, because the total misclassification error of the majority class biases the decision boundary in favor of the majority class by almost masking the minority class.[6] In such cases the classification would tend to over adapt the classes with high number of samples ignoring the smaller classes.[7] To overcome the problem of document classification involving imbalanced and low training dataset, we have implemented a solution by adding pseudo-synthetic data to the minority class.

The pseudo-synthetic data has been collected from well-known[8] taxonomical repositories and articles, and fed into the respective classes by generating files having chunk of dataset within them. In this paper we analyze the accuracy of SVM classifier with data imbalance problem in classes and also describe how pseudo-synthetic data addition to the training data set have improved the accuracy of the classifier.

1.1 Review of approaches in document classification

Text classification is considered as one of the key methods for handling and organizing text data.[9] In the study,[9] documents which generally contained strings of characters, were transformed into a suitable representation useful for learning algorithms and classification problems. Based on information retrieval research suggestions that word stems work very well as representation units leading to attribute-value representation of the text, the authors represented each distinct word along with the number of times word occurs in the document as a feature. The word stems are derived from the form of a word by removing case and derived information.[10] For example "machine", "machining" are all mapped to the same stem "machin". This led to an attribute value representation of the text data. For each unique word, the feature was represented as term frequency (ω_i, d) where, ω_i characterized a word count in the data and d characterized the corresponding text document. The stemming of the words reduced unrealistic feature variations, increased frequency count for the important feature and reduced the input-features dimension. This reduction of the curse of dimensionality[1] resulted in improved classification accuracy. To overcome the unnecessarily large feature vectors, words are represented as features if they have occurred in the training data set usually at least 3 times. Based on this representation, scaling the dimensions of the feature vector with their respective inverse document frequency (IDF, which is applied as the log inverse of ω_i) led to an improved performance. According to the study,[2] IDF can be calculated from the total number of training documents (n) and the document frequency of the particular word ω_i as shown in (1):

$$IDF(\omega_i) = \log(\frac{n}{DF(\omega_i)}) \tag{1}$$

where, DF(ω_i) is the number of those documents in the collection, which contains the term ω_i. Based on the standard feature vector representation of the text data, it was argued in the study[2] that the support vector machines are more appropriate for this type of setting. Different classification methods such as Bayes, SVM, C4.5 and kNN were applied on the Reuters-21578 and Ohsumed corpus[2] among which SVM was found to have superior prediction with considerable performance gain.

In the study,[11] authors suggest that a document can be represented as an array of words, however, not all words in a

document can be used for training the classifier, and those words to be ignored are called stop words. For example, auxiliary verbs, conjunctions and articles are classified as stop words. In a preprocessing task, stop words are often removed from consideration. In the article,[11] stop words, consisting of list of common words, were used to ignore the irrelevant features. Because of proven significance,[10] stop words and *term frequency-inverse document frequency* (TF-IDF) preprocessing techniques are implemented in our project.

Feature transformation differs significantly from feature selection approaches however, like feature selection, its main purpose lies in reduction of feature set size. Principal component analysis (PCA) has been widely used for the feature transformation.[12] The aim of the usage of PCA is to generate a discriminative transformation matrix to reduce the initial feature space into a lower dimensional feature space which reduces the complexity of the classification without any (zero to a little) trade off in accuracy as a choice. Feature extraction or feature selection techniques like term frequency, inverse document frequency and information gain described in this paper[12] are followed in the preparation of our data.

Machine learning for text classification is becoming the most popular for document classification, news filtering, document routing, etc.[13] In text classification, effective feature selection is very important to make the learner model effective and more precise. Commonly known metrics such as Chi Squared, Bi-normal Separation, Information Gain, Probability Ratio, etc. have been applied on the data to extract the features. Information Gain (IG) measures the decrement in entropy when the feature is given versus the feature is absent. IG is commonly employed as a term-goodness criterion.[14] By knowing the presence and absence of a term in a document, IG measures the number of bits of information obtained for the category prediction. Let $\{c_i\}_{i=1}^{m}$ denote the set of categories in target space, and the information gain of a term t is calculated as in (2):

$$G(t) = -\sum_{i=1}^{m} P_r(c_i)\log P_r(c_i) + P_r(t)\sum_{i=1}^{m} P_r(c_i/t)\log P_r(c_i/t) + P_r(\bar{t})\sum_{i=1}^{m} P_r(c_i/\bar{t})\log P_r(c_i/\bar{t}) \quad (2)$$

where, m is the target space dimension, and P_r is the probability of occurrence of term t in document corpus. In the seminal work of Yang and Pedersen,[14] the information gain for each unique term was computed for a given training corpus and the terms were removed from the attribute space, if the information gain was less than the predetermined threshold value. Furthermore, the authors performed evaluation of feature selection techniques like IG, Chi-squared, Document frequency thresholding, and Mutual information[14] using Reuters[15] and Ohsumed[16] data and found Chi-squared and IG were more effective in aggressive term removal without losing categorization accuracy.

For a few machine learning approaches and text classification strategies,[17] data preprocessing was done using feature extraction and feature selection approaches. In feature extraction, stemming, tokenization, stop words removal etc were applied on the documents. More than 100 variants of five major feature selection methods were tested using four popular classification algorithms: Naive Bayesian (NB) approach, Rocchio's-style classifier, kNN technique and Support Vector Machines. Among all the machine learning algorithms, Support Vector Machine, Naïve Bayes, kNN and their hybrid approaches with a combination of other algorithms have shown better accuracy in the existing literature. NB algorithm performed well in spam filtering and email classification or categorization, which requires a small amount of data for training to calculate the parameters useful for classification. On the other hand, SVM method was able to effectively collect the inherent characteristics and embed the structural risk management principle which reduced the upper bound on the generalization error. However, the main disadvantage of SVM lies in difficulty of parameter tuning and selection of kernels.

As indicated by Simon Tong,[10] support vector machines SVMs have gained remarkable success in solving number of real world problems. Usefulness of SVM based methods[18] has resulted in its widespread popularity. In the study,[10] authors represented the document as a stemmed, TF-IDF-weighted word frequency vector. According to Kim,[19] SVMs have been perceived as a standout amongst the best classification strategies for some applications including text categorization.[20] Their experimental data consists of a subset of MEDLINE database[21] with 5 categories. Each category has 500 documents which were further divided into 1,250 training documents and 1,250 test documents. The authors have proposed, two Centroid based algorithms for dimensionality reduction of clustered data. Based on these two algorithms, for term document matrix A, the reduced dimensional representation was achieved by transforming each document vector in the m dimensional space to a vector in the l dimensional space for some l < m. Additionally, for dimension transformation, either the dimension reducing

transformation $G^T \in R^{l\times m}$ was computed explicitly or the problem was formulated as a rank reducing approximation where the given matrix A was decomposed into two matrices Y and B i.e.

$$A \approx BY \tag{3}$$

where, $B \in R^{m\times l}$ with rank (B) = l, $Y \in R^{l\times n}$ with rank (Y) = l and n is the number of training samples. Moreover, the decision rule for support vector machines was formulated as:

$$y(x,j) = \text{sign}(\sum_{x_i \in SV} \alpha_i y_i K(x,x_i) + b - \theta_j^{SVM}) \tag{4}$$

where, $y(x,j) \in \{+1,-1\}$ is the classification for the document x with respect to the class j, SV is the support vectors set, and θ_i^{SVM} is the class specific threshold for the binary decision. The dual formulation of soft margin SVMs with a kernel function K and control parameter C was represented as:

$$\begin{aligned} &\alpha_i^{\max} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j K(x,x_i), \\ &s.t. \sum_{i=1}^{n} \alpha_i y_i = 0, 0 \le \alpha_i \le C, i = 1, \cdots n \end{aligned} \tag{5}$$

where, the α_i's are non-negative coefficients determined numerically and the kernel function was represented with $K(x_i,x_j) =< \phi(x_i), \phi(x_j) >$, where <,> represents an inner product of two vectors, introduced to handle nonlinearly separable cases without any explicit knowledge of the feature mapping $\varnothing$. In equation (4) the θ_i^{SVM} threshold was set so that a new document x was not classified to belong to class j when it was located very close to the optimal separating hyper or affine plane. After performing classification using algorithms like SVM and kNN, it was shown that the classification accuracy of the SVM was higher and it achieved a significant reduction in the time and space complexity. Despite the fact that the learning capacity and computational complexity of training in support vector machines might be independent of the dimension of the feature space, it has been shown that diminishing computational complexity is a fundamental issue to proficiently deal with a large number of terms in solving the text classification problems. Novel dimension reduction strategies to lessen the dimension of the text vectors drastically were implemented. Decision functions for the classification algorithm including support vector classifiers to deal with the classification issue where a text document may have the probability of belonging to multiple classes were presented. The significant exploratory results in this paper and some additional papers[19,22] demonstrate that higher efficiency for both training and testing can be accomplished without giving up prediction accuracy of text categorization even when the dimension of the input space is essentially decreased.

1.2 Overcoming data imbalance issue

According to Yanling Li, et al.,[7] imbalance in training dataset means a significant difference in the number of samples present in the classes. In such cases, the classification would tend to over adapt the classes with high number of samples ignoring the smaller classes. In most of the text classification problems this data imbalance is common. For example in monitoring the public opinion data, information security and supervision, most of the cases the quantity of texts which holds a negative view will be very low compared to the positive views. Classification algorithms like SVM, kNN, and neural network are not adaptive in handling the imbalanced data sets. Their paper mainly focused on analyzing different forms of data imbalances including text distribution, class overlap, and class size and devised the conclusions based on experimental values.

According to Rehan,[23] SVM has been widely considered and has indicated noteworthy accomplishment in numerous applications. The accomplishment of SVM is extremely constrained when it is applied to the problem of learning from imbalanced datasets in which negative cases intensely dwarf the positive cases (e.g., in gene profiling and distinguishing credit card fraud). Under-sampling the majority instances and oversampling the minority instances were the main approaches applied for imbalanced data sets. A new algorithm which needs to bias SVM in a way that will push the boundary away from the positive instances is introduced. The performance of proposed algorithm on UCI datasets[24] against the Synthetic Minority Oversampling Technique (SMOTE) algorithm by Chawla et al., consolidated with Veropoulos et al.'s[25] different error costs calculation, alongside undersampling and general SVM, demonstrate that the proposed algorithm works well. The main problem with imbalanced data sets is that they skew the boundary towards the minority class instances. The classification function for the hard margin linear SVM is as follows:

$$\text{sign}((w.b) + b) \tag{6}$$

where, w is a vector that is normal to the separating hyperplane. The norm of w and variable b decide the distance of the hyperplane from the origin. This skew of the learning hyperplane is tested on UCI datasets.[24] Ideal boundary for

this dataset is measured by testing balanced datasets which are linearly separable and noise free in the feature space. When SVM classifier is applied on the balanced UCI data, 100% accuracy is achieved. Upon repeating the experiment by keeping the positive instances same and reducing the negative instances, the results have shown a significant angles between ideal and learned hyperplane. Thus, this motivates us to reduce the data imbalance issue in our training set, while applying SVM with a good faith.

There are several methods to generate synthetic data that follow distribution of real data such as adding Gaussian, or other suitable noise[26] to the existing actual data and using subrogation techniques.[27] The quality of the subrogation techniques for multivariate data can be guaranteed by constraining them to have the same covariance, marginal distribution, and joint distributions as the real multivariate data.

2. APPROACHES TO DOCUMENT CLASSIFICATION USING MACHINE LEARNING

In supervised approach, single-label documents are those which are classified into one class only, multi-label documents are those which are classified into more than one class.[2] In this project we perform multi-class classification, in which a file is predicted into one of the predefined categories. Supervised learning models are widely applicable and can offer the insight about how the explanatory variables are related to the categorical response variable.

We applied five machine learning methods: k-Nearest Neighbors (kNN), two variations of Support Vector Machine (SVM), and two tree-based classification algorithms Random Forest and J48 on the experimental data[4] to carry out the initial classification. We have assessed the accuracies of the initial classification with 5 fold CV. Table 1 shows 5 fold CV accuracies obtained from various methods using the full text documents. Based on the results from Table 1, we have proceeded with SVM classifier algorithm to make the classification tool more robust for this experimental setup, because, in our preliminary assessment (see Table 1) SVM already has outperformed others.

Applied methods and integrations

Our experimental setup consists of the 4 stages: i) Experimental Data Collection, ii) Data Preprocessing, iii) Applying Machine Learning algorithms, and iv) Pseudo-synthetic Data Injection.

Table 1. Comparison of five machine learning methods 5 fold cross-validataion accuracy for the given 15 categories

Method	kNN (w/k = 9)	SVM (w/RBF-kernel)	J48	Random Forest	SVM (w/PolyKernel exp(1.0))
Accuracy	54%	55.4%	57.07%	61.04%	**69.2%**

Note. Values in **Bold** indicates highest accuracy

Figure 1. Steps involved in text classification process. The Experimental data is preprocessed using unsupervised and supervised filtering techniques. Machine learning algorithms are applied on the preprocessed data. Based on the model accuracy and training samples available in each class, pseudo-synthetic data is added and the process is repeated from sub-section 2.2.

The Experimental data is collected and preprocessed using StringToWordVec and AttributeSelection filters using weka class.[28] Once the data is preprocessed, the model is trained with the data. Depending on the accuracy of the model obtained using cross fold validation, pseudo-synthetic data is added to the required classes and the process is repeated from sub-section 2.2. Figure 1 shows the process flow of our methodology.

2.1 Experimental data collection

By manual processing, a team of NASA experts identified 15 top-level categories for their patents. The information about these patents is available online.[4] By selecting one of these categories, the NASA's (intelligent properties) portfolio is capable of restricting the results within the chosen category.

However, improvement is needed for the initial classification. For example, the precision of the search results of "social network" is only 4.2%; out of 24 output items (descriptions of patents), 23 are irrelevant but included due to the word "network". On the other hand, the relevant item is catego-

rized under "Health, Medicine and Biotechnology". Finding the relevant item through category-by-category search will take time because the logical relationship between "social network" and "Health, Medicine and Biotechnology" is not obvious. The problem of this example can be effectively solved by refined categorization and semantic tagging.

In the experimental data, each patent document was assigned category and was classified into 15 different categories. Table 2 shows the available training document for each class. We assumed that majority of the manually assigned classes to the patent files were correct, but some files could be misclassified. Further, some files could be qualified for more than one classes but only one class has been assigned.

Table 2. Number of training documents available in the experimental data

SL.	Classes	Patent File Count/Class
1	Aeronautics	87
2	Communications	29
3	Electrical and Electronics	47
4	Environment	28
5	Health Medicine and Biotechnology	83
6	Information Technology and Software	81
7	Instrumentation	33
8	Manufacturing	35
9	Materials and Coatings	190
10	Mechanical and Fluid Systems	73
11	Optics	71
12	Power Generation and Storage	25
13	Propulsion	24
14	Robotics Automation and Control	67
15	Sensors	192

Challenges faced in experimental data

(1) Data Imbalance Issue

Problem: Data Imbalance issue: The number of training samples for each class in the experimental data set is not consistent. For example, the samples count of "Sensors" class is 8 times more than the "Power Generation and Storage" class. Solution: The classifier model tends to favor the "Sensors" class due to being a majority class over the other class. To resolve this issue we added pseudo-synthetic data to the minority class, which contain relatively less number of training samples. Also, when the data sets are very small, this approach helps in increasing the training material for the model i.e., the number of features helpful for the classification is increased and also the classifier will be able to predict the unseen patents correctly due to addition of new and relevant words.

(2) Presence of Less Informative Sections in Patent Files

Problem: The patent file has "References" section in a patent document, which does not provide positive information about the document.

Solution: Every section of patent file in the experimental data is not useful for training. Hence, extracting features from the "References" section of a patent document merely increases the dimensionality of the space rather than providing useful information. Thus some sections like, "References", "See also" etc. were removed from the files, which deliberately provides useful words for training the classifier.

2.2 Data preprocessing

Creating the input file: The text documents in the experimental data were converted into Attribute-Relation File Format (ARFF) format. An ARFF file is an ASCII text file that describes the list of instances sharing a set of attributes. ARFF files have two different sections. First section of the ARFF file consists header information and the second section contains data information. Once the patent files are converted into ARFF file, it is preprocessed and used for preparing training file for the classifier.

At this stage, we apply filtering. This stage processes the texts of the extracted text-database into a set of unigrams (words) in two steps: i) Unsupervised filtering and ii) Supervised filtering.

Table 3. Supervised filtering options applied on data in the experimental setup

Filter Option	Value set	Explanation
wordsToKeep	1,000	This option selects 1,000 words from each category as attributes.
outputWordCounts	True	Absence/Presence of word is calculated using 0/1 format.
normalizeDocLength	True	It normalizes the frequencies of word in a document.
TFTransform	True	TFTransform is set true for the calculation of word weight.
IDFTransform	True	IDFTransform is set true for the calculation of word weight.
Stemmer	Snowball stemmer	Snowball Algorithm is selected for this option.
Stop words	Words From File	File containing stop words list is selected for this option (It gives the advantage of adding stop words manually into the file).
tokenizer	.,; : ' " ( ) ? !	WordTokenizer is selected, such that the document is converted into tokens based on the options.

First, unsupervised filter StringToWordVector[28] is applied on the data, which converts string attributes containing individual document-text, into a set of attributes representing the word occurrence information. StringToWordVector filtering options are shown in Table 3. The number of attributes generated after applying StringToWordVector filter is 4,855 words. Once the StringToWordVec filter is applied on the data, the attributes are generated without considering any relationship with respect to the class information as being unsupervised filtering. To view the attributes in sorted order of information gain, Attribute selection filter is applied on the generated attributes from unsupervised filter.

Second, we applied supervised filtering. Table 4 shows the options applied in AttributeSelection filter. The number of attributes generated after applying Attribute Selection filter is 1,218. Table 5 shows the weights and attributes after applying supervised filter on the data. The attributes are ranked in descending order of their TF-IDF weights.

Table 4. Supervised filtering options applied on data in the experimental setup

Filter Option	Value set	Explanation
Evaluator	InfoGainAttributeEval	Attributes are selected based on the information gain ratio.
Search	Ranker	Threshold value is set to 0.0, such that all attributes with positive information gain is considered.

Table 5. Weight (TF-IDF) - Attributes after supervised filtering

Weight (TF-IDF)	Attribute Name
0.1800	light
0.1660	materials
0.1637	metal
0.1571	signal
0.1437	optical
…..	…..
0.0508	electromagnetic
0.0478	suspension
0.0476	controllers
0.0471	molecule
0.0465	reflective
…..	…..
0.0242	nanotechnology
0.0220	biomolecule
0.0215	alkali
0.0214	recycling

2.3 Applying machine learning algorithms for classification

We used NASA's predefined patent documents as training data for performing classification. Implementation steps are discussed in the following section.

Applying algorithms on experimental data: The results from Table 1 suggest that the current data, without refinement cannot be used to build up a robust classifier. For example, kNN is a simpler method and the poor accuracy (54%) indicates that there exist many overlapped samples or, mislabeled samples.

On the other hand, SVM with various parameters, can very efficiently fit the (augmented) dataspace, once the given sample is considered the ground truth, which is however not true in this case and the accuracy ($\sim$70% using SVM) is not very high. Further, the tree based approaches (Random Forest, J48) which are usually good for noisy data are not even providing good accuracy either. Therefore, we planned to identify the good sample to determine the natural and effective class boundaries using simpler method. The simpler method, such as kNN, may help us avoid overfitting.

Robust Training and Testing: To arrange robust and effective training using kNN approach, we implemented the following two steps:

1) Identify the best k value in the kNN approach (see Tables 6 and 7)

2)Iteratively retrain and remove the misclassified sample and obtain best 5 fold CV accuracy

Table 6. Comparison of 5 fold CV accuracies of kNN with different values of k from 1 to 10

K value	1	2	3	4	5	6	7	8	9	10
Accuracy%	60.11%	54.98%	55.63%	55.82%	54.70%	54.98%	55.35%	54.14%	53.96%	55.07%

Table 7. Comparison of 5 fold CV accuracies of kNN with different values of k from 11 to 20

K value	11	12	13	14	15	16	17	18	19	20
Accuracy%	53.68%	54.52%	54.70%	54.33%	54.98%	54.70%	54.52%	54.24%	52.84%	53.86%

We exhaustively computed the classification accuracy, for k = 1 to 20, and there is lesser variations among the outcomes. We avoid selecting k = 1, because k = 1 by nature mostly over-fits, hence avoidable. Thus, we selected the most reasonable as well as optimal value, k = 7.

Iteratively retain and remove the miss-classified sample and obtain best 5-fold CV accuracy: This is the most important step to retain the best samples and retrain using appropriate sample for higher accuracy. For each iteration, we had to remove the misclassified files applying semi-automatic approach. The confusion matrix and the accuracy in each steps are shown below. It is to be noted that the accuracy is usually expected to be increasing in each steps - however, it can degrade while there are very insufficient sample for a class or classes, or, the samples among classes become heavily imbalanced:

Iteration#1: kNN (k = 7), accuracy 81.47%, this is achieved by removing miss-classified samples 1st time (shown in Table 8).

Iteration#2: kNN (k = 7), accuracy 89.73%, this is achieved by removing misclassified samples 2nd time (shown in Table 9).

Table 8. Confusion matrix of the samples kNN Iteration 1

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	<-- classified as
50	0	2	0	0	1	0	0	0	3	0	0	1	0	0	a = Aeronautics
1	15	2	0	0	0	0	0	0	0	0	0	0	0	0	b = Communications
1	3	19	0	0	0	0	0	0	0	0	0	1	0	0	c = Electrical and Electronics
0	0	0	13	0	0	0	0	0	1	0	1	0	0	0	d = Environment
4	0	3	0	46	0	0	0	1	0	1	0	0	0	1	e = Health Medicine and Biotechnology
5	2	1	0	0	29	0	0	0	3	0	0	0	0	0	f = Information Technology and Software
1	1	0	0	0	0	7	0	0	0	0	0	0	0	0	g = Instrumentation
0	0	0	0	0	0	0	21	0	0	0	0	0	0	0	h = Manufacturing
1	0	3	0	0	0	0	0	115	3	0	0	0	0	1	i = Materials and Coatings
5	0	0	0	0	0	0	0	0	48	0	1	0	0	1	j = Mechanical and Fluid Systems
1	1	2	0	0	0	0	0	0	3	21	0	0	0	2	k = Optics
0	1	0	0	0	0	0	0	0	0	0	4	0	0	0	l = Power Generation and Storage
1	0	0	0	0	0	0	0	0	1	0	0	17	0	0	m = Propulsion
3	0	0	0	0	0	0	0	0	2	0	0	0	39	0	n = Robotics Automation and Control
6	4	3	1	0	0	0	0	2	12	13	0	0	0	53	o = Sensors

The confusion matrix shown in Table 10, is achieved by performing SMO poly-kernel method on kNN's second iteration data. The accuracy achieved is, 96.0331%. It is to be noted, that as the training samples further decreased in classes like "Communications", "Power Generation and Storage" etc. we restored the original samples into the data set to avoid poor training set. To improve the model accuracy further we have proposed to add pseudo-synthetic data to the training files which is discussed in the following section.

Table 9. Confusion matrix of the samples kNN Iteration 2

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	<-- classified as
47	2	0	0	0	0	0	0	0	0	0	0	1	0	0	a = Aeronautics
0	14	0	0	0	0	0	0	0	0	0	1	0	0	0	b = Communications
1	0	17	0	0	0	0	0	0	0	0	0	1	0	0	c = Electrical and Electronics
0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	d = Environment
6	3	1	0	33	0	0	0	0	1	2	0	0	0	0	e = Health Medicine and Biotechnology
3	2	0	0	0	23	0	0	0	0	1	0	0	0	0	f = Information Technology and Software
0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	g = Instrumentation
0	0	2	0	0	0	0	18	0	1	0	0	0	0	0	h = Manufacturing
1	0	3	0	0	0	0	2	109	0	0	0	0	0	0	i = Materials and Coatings
4	0	2	0	0	0	0	0	0	42	0	0	0	0	0	j = Mechanical and Fluid Systems
0	1	1	0	0	0	0	0	0	0	19	0	0	0	0	k = Optics
0	0	2	0	0	0	0	0	0	0	0	2	0	0	0	l = Power Generation and Storage
0	0	0	0	0	0	0	0	0	0	0	0	17	0	0	m = Propulsion
0	0	0	0	0	0	0	0	0	1	0	0	0	38	0	n = Robotics Automation and Control
0	1	2	0	0	0	0	0	1	0	2	0	0	0	47	o = Sensors

Table 10. Confusion matrix of the samples kNN Iteration 2 while using SVM with poly-kernel

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	<-- classified as
46	1	0	0	0	0	0	0	1	2	0	0	0	0	0	a = Aeronautics
0	28	0	0	0	0	0	0	0	0	0	0	0	0	2	b = Communications
0	0	26	0	0	2	0	0	2	0	0	0	0	0	0	c = Electrical and Electronics
0	0	0	26	0	0	0	0	0	0	0	0	0	0	0	d = Environment
0	0	0	0	45	0	0	0	0	0	0	0	0	0	1	e = Health Medicine and Biotechnology
1	0	0	0	0	27	0	0	0	0	1	0	0	0	0	f = Information Technology and Software
0	0	0	0	0	0	28	0	0	0	0	0	0	0	0	g = Instrumentation
0	0	0	0	0	0	0	30	0	0	0	0	0	0	0	h = Manufacturing
0	0	0	0	0	0	0	0	114	1	0	0	0	0	0	i = Materials and Coatings
1	0	1	0	0	0	0	0	1	45	0	0	0	0	0	j = Mechanical and Fluid Systems
0	0	1	0	0	0	0	0	0	0	26	0	0	0	0	k = Optics
0	0	0	0	0	0	0	0	0	0	0	28	0	0	0	l = Power Generation and Storage
0	0	0	0	0	0	0	0	0	0	0	0	26	0	0	m = Propulsion
0	0	0	0	0	0	0	0	0	1	0	0	0	38	0	n = Robotics Automation and Control
1	1	0	0	0	0	0	0	3	0	0	0	0	0	48	o = Sensors

2.4 Addition of pseudo-synthetic data in training

Reducing the difference in number of training samples between classes: We first tried to increase the number of samples in the less populated classes (i.e., classes having fewer number of training samples) by copy pasting the available files in their respective classes. Doing so will increase the count of files in the classes but could not achieve much accurate results upon testing. As the samples in small classes are copy-pasted, it increases the number of training data and increases the weight of the training attributes. However, as we tested the data, which is not seen by the training model, it resulted in poor accuracy. And the accuracy was 46.2366%

and total 93 files tested.

To overcome the problem of text classification involving imbalanced and low quantity of training dataset, we have implemented a solution by adding pseudo-synthetic data in the classes that have less number of samples.[30] The pseudo-synthetic data has been collected from well-known taxonomical repositories[8] and articles, and fed into the respective classes by generating files having chunk of dataset within them. Keywords related to each class has also been collected in this procedure and are added into text files and are fed into the classes for training purpose. The purpose of the addition of the keywords is to have higher information gain for the keywords/attributes, which are very closely related to a particular class. In future when a new patent document is given for prediction, it is expected that due to feeding relevant data in the training pseudo-synthetically, the prediction can be made more accurate. Pseudo-synthetic data is collected based on the major areas that are sub domains for the top class. For example, for the top class, "Aeronautics", we have collect data that is suitable for sub-categories such as: "Aerodynamics and Fluid Mechanics", "Structures and Materials", "Propulsion and Power", etc.

We trained the model with the relevant data and tested it with several other related input files. Based on the data provided by NASA and the pseudo-synthetic data that is collected, sub categories are classified. We read the documents thoroughly and cross-validated against mature resources like IEEE[8] for the sub category classification and fit the data in the respective sub classes for training.

2.4.1 *Pseudo-synthetic data collection*

We have collected data from MediaWiki database. The main advantage of adding pseudo-synthetic data is that the model can handle new/unseen data that comes for testing hence can result in better prediction. By adding pseudo-synthetic data into classes, the ill impact of imbalance has been subsided and the domination of the higher sample classes reduces gradually, which ultimately improves the model accuracy for testing cases, i.e., true prediction in classification enhanced.

2.4.2 *Web crawler for data collection*

Collecting data from each source is a tedious task. This issue can be resolved by using web crawling technique. A web crawler is a simple program to collect data from web, which uses web links as inputs. We built a web crawler to collect data from internet (MediaWiki) and use it in training data set. We have collected sub category list for each top category from IEEE taxonomy. We provide web crawler the URL of MediaWiki and a taxonomy word to search for. The crawler will go to the web page and download as text file and if the link doesn't work, it will go to the next page and repeat. Visiting the pages that is already visited should be taken care of while extracting data. Thus the data that is required to reduce the class imbalance issue is collected.

Table 11 shows the increase in file count for each category using web crawling technique. The training samples increased from 1,065 to 1,600, which reduced the data imbalance issue in the training dataset.

Table 11. Training sample file count before and after insertion of pseudo-synthetic data

SL.	Classes	Old File Count/Class	New File Count/Class
1	Aeronautics	87	125
2	Communications	29	52
3	Electrical and Electronics	47	93
4	Environment	28	44
5	Health Medicine and Biotechnology	83	98
6	Information Technology and Software	81	105
7	Instrumentation	33	130
8	Manufacturing	35	121
9	Materials and Coatings	190	192
10	Mechanical and Fluid Systems	73	99
11	Optics	71	112
12	Power Generation and Storage	25	73
13	Propulsion	24	24
14	Robotics Automation and Control	67	118
15	Sensors	192	214
Total		**1065**	**1600**

3. RESULTS

The performance of five of the methods implemented in this work is evaluated based on an independent test dataset (TsD) of 116 randomly selected documents, belonging to various classes, from the dataset, which was generated after second iteration of kNN (see Table 6). Two separate training datasets are prepared: i) dataset before addition of pseudo-synthetic data (TrD1) and ii) dataset after addition of pseudo-synthetic data (TrD2). TrD1 is the same dataset obtained after second iteration of kNN (see Table 6). None of the documents from the independent test dataset were used in the training of the models. Figures 2, 3 and 4 show the performance comparison of five different methods based on three different indices: i) overall accuracy, ii) kappa statistics and iii) recall score respectively. All of the five machine learning methods (kNN, SVM [w/RBF-kernel]), J48, Random Forest and SVM [w/PolyKernel]) were separately trained using two separate

training dataset TrD1 and TrD2 then tested on an independent dataset TsD. The first performance measure, overall accuracy, provides the percentage of correctly classified instances. It is computed as the number of correctly classified instances over the total number of instances present in the dataset. The second measure, kappa statistics, is computed as the difference between observed agreement and expected agreement. It is standardized to have value in between -1 to 1, where 1 is perfect agreement, 0 is exactly what would be expected by chance, and negative values indicate agreement less than chance. The third measure, recall score (also known as sensitivity), is used to identify a classifier's completeness in classifying the positive class. It is computed as the proportion of instances classified as a given class divided by the actual total in that class.

From Figure 2, it is evident that SVM with poly-kernel method outperforms kNN, SVM with RBF-kernel, J48 and Random Forest methods based on overall test accuracy, regardless of whether the methods are trained with TrD1 or TrD2. Moreover, Figure 2 indicates that SVM with RBF-kernel, J48 and SVM with poly-kernel methods provide improved performance when the models are trained using TrD2 instead of TrD1. Besides, kNN and Random Forest methods show reduced accuracy when the models are trained using TrD2. Likewise, from Figure 3, we can see that SVM with poly-kernel method performs better than kNN, SVM with RBF-kernel, J48 and Random Forest methods based on kappa statistics, regardless of whether the methods are trained with TrD1 or TrD2. Additionally, Figure 3 indicates that SVM with RBF-kernel, J48 and SVM with poly-kernel methods provide improved performance when the models are trained using TrD2 whereas, kNN and Random Forest methods show reduced accuracy. In a like manner, the results indicated by Recall score in Figure 4 shows similar trend to the results indicated by overall accuracy and kappa statistics in Figures 2 and 3. Thus, we can see that all the three indices we used for performance evaluation suggests that injection of pseudo-synthetic data has helped improve the accuracy of the classifier.

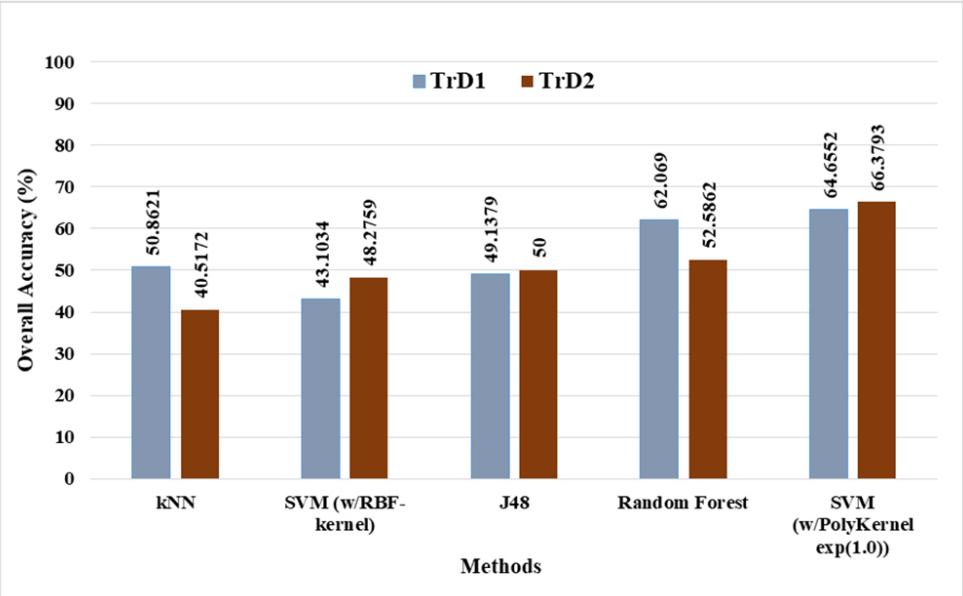

Figure 2. Comparison based on overall accuracy of five different machine learning methods trained using two different training datasets (TrD1 and TrD2) and tested on TsD

Furthermore, using TrD1 for training, TsD for testing and SVM with poly-kernel based machine learning method, we achieved highest overall accuracy, kappa statistics and recall score respectively 64.6552%, 0.6144 and 0.647. On the other hand, using TrD2 for training, TsD for testing and SVM with poly-kernel based machine learning method, we achieved the highest overall accuracy, kappa statistics and recall score, respectively, 66.3793%, 0.6319 and 0.664. Thus, there is a clear improvement of about 3% in test accuracy when TrD2 is used to train SVM with the poly-kernel method. Tables 12 and 13 show the confusion matrix obtained using SVM with poly-kernel method, where TrD1 and TrD2 are used as the training sets respectively and TsD is used as the test set. In addition to confusion matrix, Figures 5 and 6 shows the scatter plot of actual versus predicted class for the test dataset, TsD, obtained using SVM with poly-kernel method

where TrD1 and TrD2 are used as training set respectively. The overlapped cross and square shapes shows the correct prediction. The count of overlapped shapes in each row of Figure 5 and 6 provide the number of correctly classified documents. Likewise, columns of Figures 5 and Figure 6 show the correctly classified and misclassified documents. In case, the document is misclassified, the actual and the incorrectly predicted class of the document can be seen from these figures. The visual comparison of Figures 5 and 6 show that the higher number of overlapped shapes are presents in Figure 6, which implies that the injection of pseudo-synthetic data resulted in improved performance. The reason behind the improvement of the accuracy of the model is the weights (TF-IDF) of the attributes, which assisted further information gain.

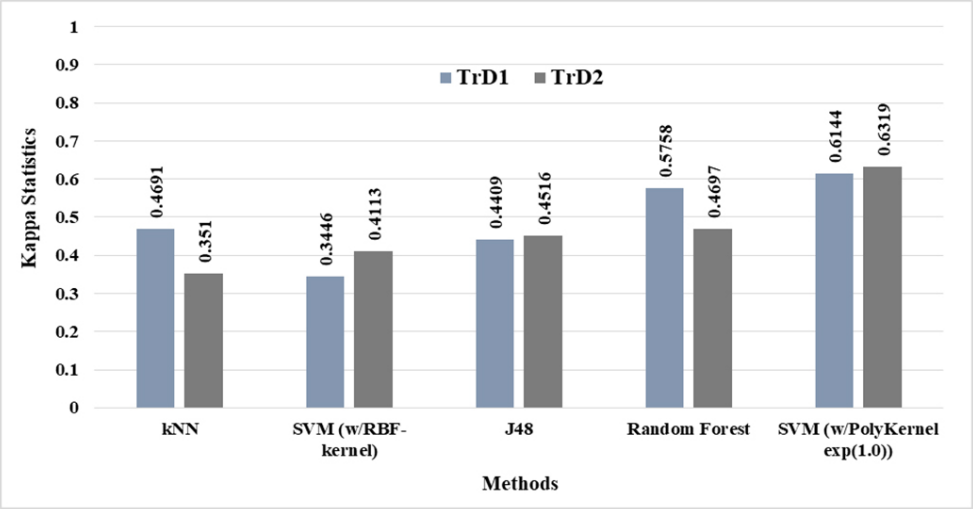

Figure 3. Comparison based on Kappa statistics of five different machine learning methods trained using two different training datasets (TrD1 and TrD2) and tested on TsD

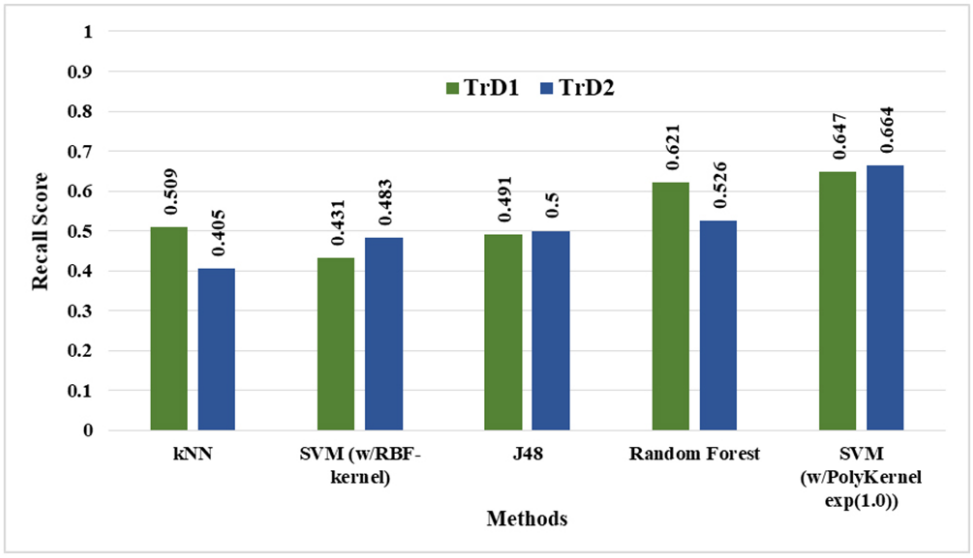

Figure 4. Comparison based on recall score of five different machine learning methods trained using two different training datasets (TrD1 and TrD2) and tested on TsD

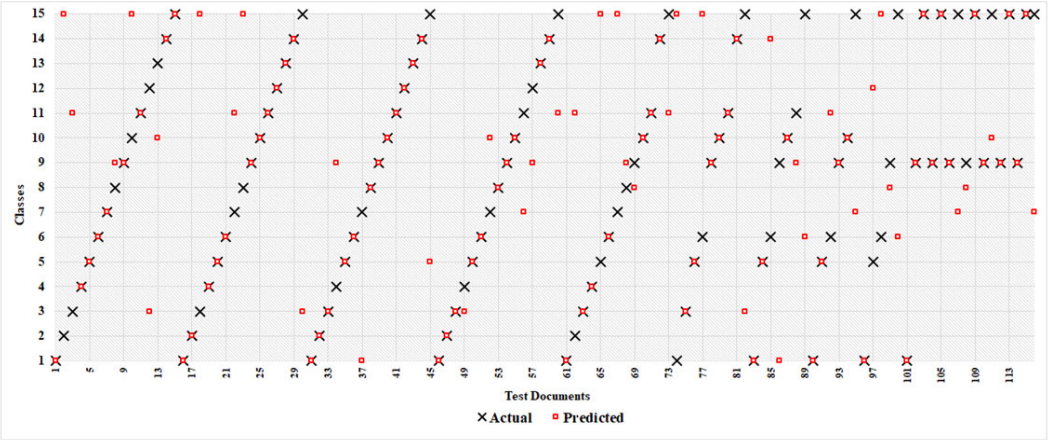

Figure 5. Scatter plot of actual versus predicted class for test dataset TsD. Here, the SVM with poly-kernel was trained using TrD1. The overlapped cross and square shape show the correct prediction.

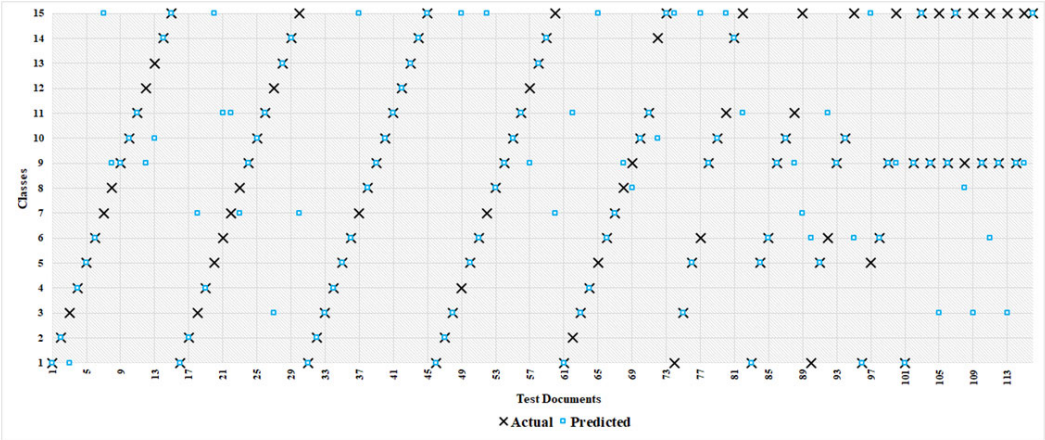

Figure 6. Scatter plot of actual versus predicted class for test dataset TsD. Here, the SVM with poly-kernel was trained using TrD2. The overlapped cross and square shape shows the correct prediction

Table 12. Confusion matrix of the test samples before addition of pseudo-synthetic data

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	<-- classified as
9	0	0	0	0	0	0	0	0	0	0	0	0	0	1	a = Aeronautics
0	3	0	0	0	0	0	0	0	0	1	0	0	0	1	b = Communications
0	0	4	0	0	0	0	0	0	0	1	0	0	0	1	c = Electrical and Electronics
0	0	1	3	0	0	0	0	1	0	0	0	0	0	0	d = Environment
0	0	0	0	7	0	0	0	0	0	0	1	0	0	1	e = Health Medicine and Biotechnology
0	0	0	0	0	5	0	0	0	1	0	0	1	2		f = Information Technology and Software
1	0	0	0	0	0	1	0	0	1	1	0	0	0	1	g = Instrumentation
0	0	0	0	0	0	0	2	2	0	0	0	0	0	1	h = Manufacturing
1	0	0	0	0	0	0	3	12	0	0	0	0	0	0	i = Materials and Coatings
0	0	0	0	0	0	0	0	0	7	0	0	0	0	1	j = Mechanical and Fluid Systems
0	0	0	0	0	0	1	0	1	0	5	0	0	0	0	k = Optics
0	0	1	0	0	0	0	0	1	0	0	2	0	0	0	l = Power Generation and Storage
0	0	0	0	0	0	0	0	0	1	0	0	3	0	0	m = Propulsion
0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	n = Robotics Automation and Control
0	0	2	0	1	2	3	0	0	1	2	0	0	0	6	o = Sensors

Table 13. Confusion matrix of the test samples after addition of pseudo-synthetic data

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	<-- classified as
9	0	0	0	0	0	0	0	0	0	0	0	0	0	1	a = Aeronautics
0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	b = Communications
0	0	4	0	1	0	0	0	0	0	0	0	0	0	1	c = Electrical and Electronics
0	0	0	3	1	0	0	0	0	1	0	0	0	0	0	d = Environment
0	0	0	0	7	0	0	0	0	0	0	1	0	0	1	e = Health Medicine and Biotechnology
0	0	1	0	0	4	0	0	0	1	0	0	1	2		f = Information Technology and Software
0	0	0	0	0	0	1	0	1	0	1	0	0	0	2	g = Instrumentation
0	0	0	0	0	0	0	2	2	0	0	0	0	0	1	h = Manufacturing
0	0	0	0	0	0	0	2	13	1	0	0	0	0	0	i = Materials and Coatings
0	0	0	0	0	0	0	1	0	7	0	0	0	0	0	j = Mechanical and Fluid Systems
0	0	0	0	0	0	1	0	1	0	4	0	0	0	1	k = Optics
0	0	0	0	0	0	0	0	0	1	0	1	2	0	0	l = Power Generation and Storage
0	0	0	0	0	0	1	0	0	1	0	0	2	0	0	m = Propulsion
0	0	0	0	0	0	0	0	0	0	0	1	0	0	5	n = Robotics Automation and Control
0	0	0	0	0	0	1	0	1	1	0	0	0	1	13	o = Sensors

4. CONCLUSIONS AND FUTURE WORK

In this paper, we designed a novel method for patent document classification and sub classification using machine learning techniques. Our method involves data preprocessing techniques necessary for document classification and handling overlapped classes. In this study, we found that the accuracy of the document classifier (5 fold cross validation results) improves from 69.20% to 96.03% with the removal of misclassified files from the training dataset. In addition, data imbalance issue causes a drop in the accuracies of the classifier models. To mitigate the problem, we applied several techniques to reduce the data imbalance issue in training dataset. Fortunately, the addition of pseudo-synthetic data improved the classification accuracy. This process might help the future researchers to proceed further in the usage of pseudo-synthetic data for the reduction of data imbalance problem in training sets in a meaningful way. We have developed two models for document classification, first, excludes the misclassified files from the training data, second, does not exclude misclassified files, where the former model achieved more accuracy percentage. It is to be noted that both the models are equally important as the class overlap helps us acquire the probabilities with which the input file fall into multiple classes. Furthermore, we suggest the method of obtaining the tag words which provide fine-grained level of details about the input file or categories generated based on the information gain of the terms. The terms with highest frequency gain from the data were considered as the tag words. We would like to draw the attention of researchers to perform in detail analysis and implementation of sub category classification, as it needs more training data for building a robust model, which can be achieved by collecting more pseudo-synthetic data.

In addition to single classifier, fusion technique could further improve the accuracy of the multiclass classification problem of patent documents, studied under this work. Potential fusion techniques could be ensembles of classifiers with voting scheme, cascaded classifiers, stacking of ensemble of classifiers, cascaded classifiers with iterative learning etc. Further, it would be interesting to see if the cascaded learners, the output of one learner, used as an input to the next learner, could help improve the prediction accuracy.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge NASA grant: NASA (2016)-Stennis-ll, for supporting this study.

REFERENCES

[1] Hastie T, Tibshirani R, Friedman J. The Elements of Statistical Learning. 2009: Springer.

[2] Wang TY, Chiang HM. Solving multi-label text categorization problem using support vector machine approach with membership function. Neurocomputing. 2011; 74(17): 3682-89. https://doi.org/10.1016/j.neucom.2011.07.001

[3] Chapelle O, Scholkopf B, Zien A. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. IEEE Transactions on Neural Networks. 2009; 20(3): 542-542.

[4] Patft. [cited 2017 March 17th]. Available from: http://patft.uspto.gov/netahtml/PTO/index.html

[5] Kullback S. Information Theory and Statistics. 1959. NY: Wiley.

[6] Longadge MR, Dongre MSS, Malik L. Multi-cluster based approach for skewed data in data mining. Journal of Computer Engineering (IOSR-JCE). 2013; 12(6): 66-73. https://doi.org/10.9790/0661-1266673

[7] Li Y, Sun G, Zhu Y. Data imbalance problem in text classification. in Information Processing (ISIP), 2010 Third International Symposium on. 2010. IEEE.

[8] IEEE Taxonomy.

[9] Joachims T. Text categorization with support vector machines: Learning with many relevant features. European conference on machine learning. Springer Berlin Heidelberg. 1998.

[10] Tong S, Koller D. Support vector machine active learning with applications to text classification. Journal of Machine Learning Research, 2001 2(Nov): 45-66.

[11] Ikonomakis M, Kotsiantis S, Tampakas V. Text classification using machine learning techniques. WSEAS Transactions on Computers. 2005; 4(8): 966-974.

[12] Zu G, et al. Accuracy improvement of automatic text classification based on feature transformation. In Proceedings of the 2003 ACM symposium on Document engineering. 2003. ACM.

[13] Forman G. An extensive empirical study of feature selection metrics for text classification. Journal of Machine Learning Research. 2003 3(Mar): 1289-1305.

[14] Yang Y, Pedersen JO. A comparative study on feature selection in text categorization. In Icml. 1997.

[15] Reuters-21578 Text Categorization Collection Data Set. [cited 2017 March 20th]. Available from: https://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection

[16] Ohsumed and Reuters text classification datasets – download. [cited 2017 March 18th]. Available from: https://www.mat.unical.it/OlexSuite/Datasets/SampleDataSets-about.htm

[17] Khan A, et al. A review of machine learning algorithms for text-documents classification. Journal of Advances in Information Technology. 2010; 1(1): 4-20.

[18] Iqbal S, Mishra A, Hoque MT. Improved prediction of accessible surface area results in efficient energy function application. Journal of Theoretical Biology. 2015; 380: 380-391. PMid:26092374. https://doi.org/10.1016/j.jtbi.2015.06.012

[19] Kim H, Howland P, Park H. Dimension reduction in text classification with support vector machines. Journal of Machine Learning Research. 2005 6(Jan): 37-53.

[20] Wang S, Manning CD. Baselines and bigrams: Simple, good sentiment and topic classification. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2. 2012. Association for Computational Linguistics.

[21] U.S. National library of Medicine MedLine dataset. [cited 2017 March 19th]. Available from: https://www.nlm.nih.gov/databases/download/pubmed_medline.html

[22] Mikolov T, et al. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. 2013.

[23] Akbani R, Kwek S, Japkowicz N. Applying support vector machines to imbalanced datasets. in European conference on machine learning. 2004. Springer.

[24] Machine Learning Repository. [cited 2017 March 19th]. Available from: https://archive.ics.uci.edu/ml/datasets.html

[25] Chawla NV, et al. SMOTE: synthetic minority over-sampling technique. Journal of Artificial Intelligence Research. 2002; 16: 321-357.

[26] Mivule K. Utilizing noise addition for data privacy, an overview. arXiv preprint arXiv:1309.3958, 2013.

[27] Salazar A, Safont G, Vergara L. Surrogate techniques for testing fraud detection algorithms in credit card operations. in Security Technology (ICCST). 2014 International Carnahan Conference on. 2014. IEEE.

[28] Smith TC, Frank E. Introducing machine learning concepts with WEKA. Statistical Genomics: Methods and Protocols. 2016: 353-378.

[29] Attribute-Relation File Format (ARFF). [cited 2017 March 26th]. Available from: http://www.cs.waikato.ac.nz/ml/weka/arff.html

[30] Barua S, et al. MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning. IEEE Transactions on Knowledge and Data Engineering. 2014; 26(2): 405-425. https://doi.org/10.1109/TKDE.2012.232