International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-1, October, 2019

A Comprehensive Framework for Ontology based Classifier using Unstructured Data

M.Thangaraj, M.Sivakami

Abstract: ***The knowledge contained within the natural language data can be used to build expert systems. Classifying unstructured data using ontology and text classification algorithms to extract information is one way of approaching the problem of building intelligent systems. One major problem with text processing is most data generated is unstructured and ambiguous, as, data with a structure helps to identify meaningful patterns and eventually exhibit the latent knowledge. Ambiguity in natural language affects accuracy of categorization. Also, Natural Language Processing techniques when combined with semantic data modeling through ontological knowledge will also solve the problem of domain knowledge representation thereby enabling improved data classification facilities, particularly in large datasets where number of features scale to unmanageable proportions. In this paper, the domain knowledge is presented as a knowledge graph, derived from the semantic data modeling. Further, to achieve better Multi Class classification, Multinomial Naive Bayes algorithm is applied to categorize items in their respective classes. For the experiments, Data about various news groups were used for testing the accuracy of the model. Experimental results have proved that the proposed classifier performs better compared to existing systems.***

Keywords : Text categorization, Multiclass classification, Ontology, Knowledge Graphs, Feature Hashing, Topic Modeling.

I. INTRODUCTION

Text analytics or Text mining, can be defined as the process of extracting patterns of interest from raw data, where the patterns in turn will become the knowledge obtained. With this knowledge, a better understanding of various problems is achieved, which is otherwise difficult for a human mind when the amount of data involved is vast. Some of the applications of text analytics are, spam e-mail classification, sentiment tagging of online reviews and tweets, knowledge extraction from historical data etc. All these applications is seen as a classification or categorization problem. Achieving a structured data by classifying raw data based on its inherent patterns is a fundamental challenge[1].

Classification can be supervised, unsupervised and semi-supervised and further, based on the number of target classes, classification is called as binary, multi-label and multi-class classification. In this paper, we have taken up multi-class classification where the number of target classes are more than two[2]. The aim is to achieve unsupervised multi class classification in document filtering. In order to attain such a classification this paper uses tools from various fields such as Natural Language Processing (NLP) [3], Machine Learning (ML) and Semantic Data Mining (SDM). From NLP, data preprocessing techniques and topic modeling techniques were incorporated. Data preprocessing removes noisy data and normalize it in a form suitable for further analysis. Topic modeling selects the most appropriate features about the entity of interest to enable precise categorization. Since the first step in any data analytics involves studying the data, it's very important to any remove unnecessary information which will otherwise consume time.

Unsupervised text classification has another major issue in the form of semantically rich data corpus. Refining the exact contextual meaning of data before assigning them to categories is a requirement to be fulfilled, especially in multi-class classification problems. Where the relation between different topics and also the relation between words within a topic are used to measure similarity between documents. To extract the relationship between topics of interest, a domain ontology is used. Ontology[4] is the formal specification of shared conceptualization which teaches a machine to learn a concept, by describing the inter-relationship between various terms related to the concept. This is another practice added in the proposed classifier to tune the accuracy despite the problems of synonymy and polysemy.

The proposed framework is elaborately designed to achieve classification that is both semantic and syntactic and also performs better without multi tier classifier algorithms. As, multi tier classifiers are the norm to improve accuracy despite the time complexity overhead. Also, ensemble classifiers, lack interpretability and highly complex[5]. To automate the task of effective document categorization, Multinomial Naive Bayes(MNB) algorithm is used. It uses Bayes Theorem to predict membership probabilities for each class such as the probability that given an entity belongs to a particular class. The class with highest likelihood is regarded as the most likely class. It is used on data that is multinomially distributed[6]. One of the drawbacks of MNB is its inability to relate features, which can be rectified by giving ontology as training data, this is the novelty of this proposed work.

In this work, we propose a novel ontology-based multi-class text classifier to extract semantic and syntactic relations between concepts from the documents corpus and categorize them according to the respective class labels. The novelty of this approach is that it does not rely on training a classifier or manually built ontologies. Instead, uses concepts and their relationships represented in the automatically generated

Revised Manuscript Received on October 30, 2019.
* Correspondence Author
Dr.M Thangaraj*, Professor ,Department of Computer Science, School of Infor- mation Technology, Madurai Kamaraj University, India.
Ms. M Sivakami, Research Scholar Department of Computer Science, School of Information Technology, Madurai Kamaraj University, India.

domain ontology. This in turn becomes a knowledge graph for the classifier model, which classifies new documents using the knowledge graph as training data.

The remainder of the paper is organized as follows: Section 2 delivers the various directions of research in this area. The proposed architecture with elaborate discussion of the working of the model was discussed in Section 3. The system was compared with benchmark classifiers to reason out the performance of the system in Section 4. Section 5 presented the conclusion and future enhancement.

II. LITERATURE REVIEW

Among some of the ways to achieve [7] automatic text classification, by using semantic similarity between the content of a text document and a section of the ontology. The notion is to exploit relationships between entities in a document to determine the document categorization. And also entities of the similar domains are very closely related. Ontology represents the data about the domain of interest along with named entities and relationships that have appropriate labels to identify them in documents. The entities will be classified based on class taxonomy of the ontology. Though the richness of domains are better displayed, with any number of entities the ontology traversal was time consuming. It considered only building manual ontology using data from Wikipedia. The categorization was only suitable for encyclopedic-type [8] ontology.

Extracting knowledge from unstructured textual data for decision making processes is highly challenging right from data collection, organization and analysis. Traditionally classification approaches [9]used statistical and machine learning methods to perform decision making. For all these methods, a training dataset or pre-classified sample documents were mandatory[10]. Only using this knowledge, the new unseen documents could be classified. Integration of semantic information into text categorization is yet to be fully explored. Earlier WordNet clusters [11]words into synonym sets to improve. Having a dictionary in RAM slowed down process completion.

The ontology-based approach is computationally demanding to identify topics that appear in text from the ontology. The classification space is further reduced according to the hierarchy to achieve generalization [12]. Each concept is treated as a set of terms in the text. Named entity recognition is again introduced to represent semantics which was not capable to avoid ambiguity. Creating the document into vector space model is another major obstacle that could be solved when introducing ontology. Further, using solely ontologies slowed down the process which was then solved by using machine learning algorithms. These algorithms not only automated the processes but also saved starting from the scratch[13].

PubMed is by far the biggest biomedical literature repository which contains 18 million plus articles and growing [14]. It is difficult to search the entire corpus to extract relevant knowledge. Contextual query[15] analysis is also tedious as each user looks at the data from a different perspective. Extracting synopsis knowledge from PubMed is time consuming and work intensive. Automatic knowledge discovery helps to attain quality work, circumvent repetition, and produce new using the older data as template. However, it required application of complex deep learning algorithms to get the desired results[16]. Training the classifier also produced time complexity issues which could not be circumvented. Obtaining the semantic knowledge about content is the major requirement when it comes to natural language data.

The feature selection techniques[17] greatly influences the performance of text classifiers. Using ontologies to represent document features is tedious, given the number of features for a particular data corpus. The ontology [18]could only be built manually in such a scenario to produce accurate classification. It first identify concepts then arrange them hierarchically to build domain taxonomy. The framework used a hierarchical softmax skip-gram algorithm similarity based approach to learn concepts. Single domain ontologies [19] are difficult to be considered for other approaches in various other domains also calls for domain experts, which is not possible in all scenario.

From the review of the literature, some of the research gaps with respect to ontology based multi class text classification mechanisms are identified. They are given below,

A. PROBLEMS IDENTIFIED:

- Manual ontology construction is time consuming and tedious
- Semantic decision support systems are required when more natural language data is generated.
- Application of Machine Learning algorithms in ontology based models has many research gaps like imbalanced nodes, requirement of multi layer graph training and data that cannot be exhibited in a relational table.
- When handling large feature sets, the classifier efficiency is reduced irrespective of the potential of the algorithm.
- Dictionary developed during topic modeling and classification is memory intensive.

This work aims to solve the above mentioned problems using NLP techniques and Machine Learning algorithms in ontology about a domain. This system can be used as decision support system that involves processing unstructured text data with rich semantic relationships.

III. PROPOSED METHODOLOGY

The proposed system is named as OntoClassifier. The basic functions include collecting unstructured data(raw text documents) from the web and extracting inherent topics. The topics and their relevant terms are represented using an ontology to preserve the semantic value of natural language data. After the ontology is obtained it becomes the training data for the Graph based data model using Machine Learning algorithm. This semantic data model proposes an improved Multinomial Naive Bayes algorithm to achieve effective Text Classification. The OntoClassifier architecture can be broadly classified into three layers: Data Layer, Ontology Layer and Classification layer. The architectural framework for the proposed classifier is given in Fig.1.

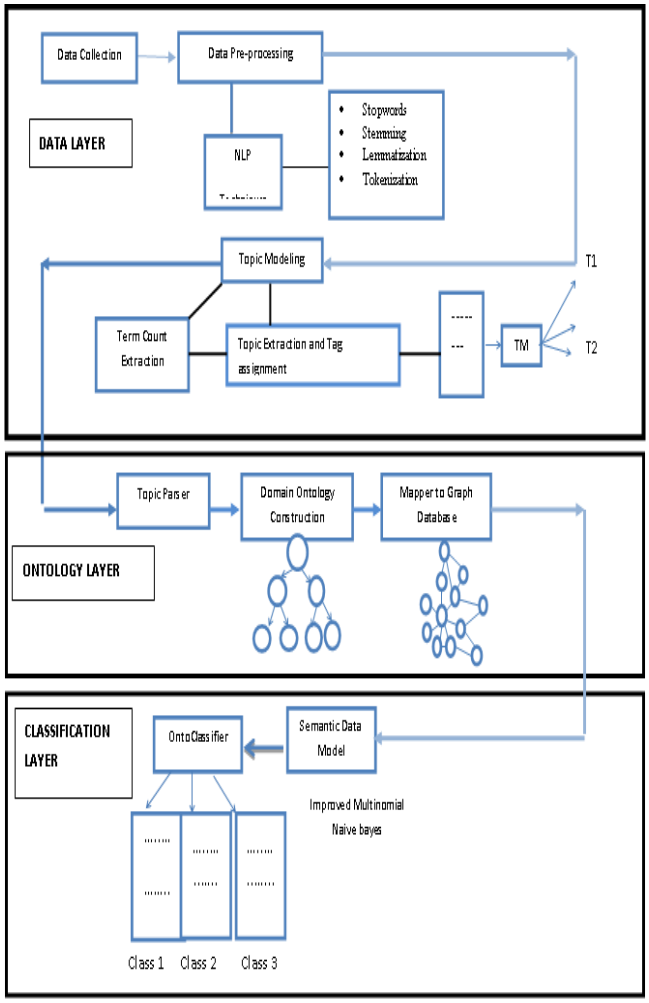


Fig. 1.Ontology based multi-class text classification framework

3.1 Data Layer

The data layer contains three modules namely, Data Collection, Data Pre-Processing and Topic Modeling. The specific functions of each module are listed below.

3.1.1 Data Collection

A news dataset which was readily available on Kaggle called '20NewsGroups' was obtained. It contains 20 categories of news generated and for each category 1000 documents amounting to 20000 news documents in textual format were downloaded. Based on the categories already available, A web crawler was constructed for Google news, BBC, Reuters and HuffingtonPost to collect daily news in document format. The crawler was constructed using scrapy library of python, it was programmed to collect 10 documents for each category on a daily basis for a month with a total of 6000 documents were crawled and organized according to their categories. Therefore, the total number of documents taken for the study is 26000 text documents.

3.1.2 Data Preprocessing

Once the dataset is imported, it is analyzed for class distribution as given in Fig.2 to inspect for class imbalance problem. Imbalance[20] class distribution gives biased results therefore it is important to ensure all the classes are well balanced and no class is heavily represented. From the Fig. 2, it is seen that most of the classes are evenly distributed. Secondly, the data obtained is in natural language form which has to be preprocessed to aid in knowledge discovery. Text normalization[21] will convert human readable data into machine readable form. Some of the normalization techniques executed for the Onto Classifier are stop words, stemming, lemmatization and tokenization.

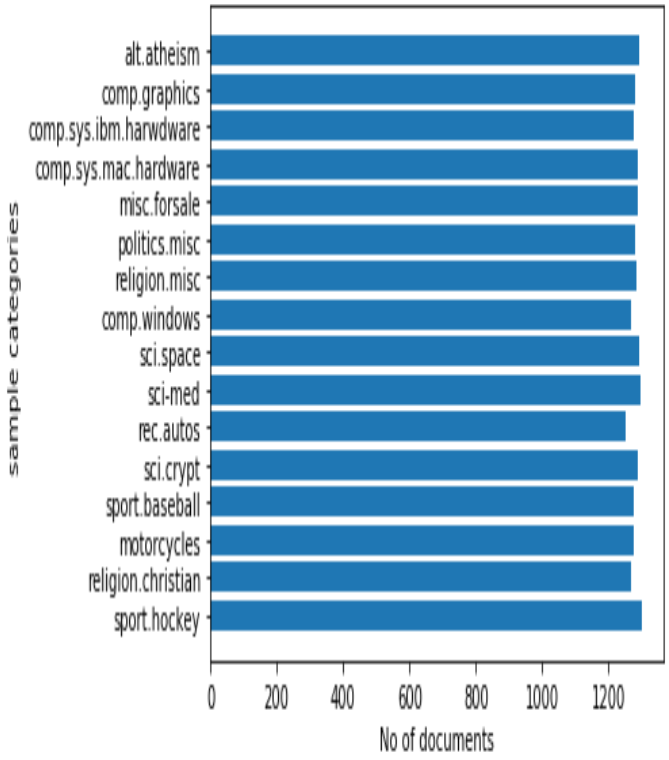


Fig 2. Class distribution of newsgroup dataset

3.1.3 Topic modeling

After the data preprocessing, it is a process to automatically identify topics present in a text object and to derive hidden patterns exhibited by a text corpus. It enables effective decision making. The algorithm used for topic modeling is Latent Drichlet Allocation(LDA).

3.1.3. a. Term Count Extraction

The terms for each topic will be grouped using the [22]Bag of Words model and a dictionary will be populated. This dictionary will also be used in a different place apart from its usage LDA. There are two inputs to the topic model one is the dictionary and the other one is corpus. Gensim a python library for automatic topic modeling technique assigns a unique identifier for each word from the document. For example, (0, 2) implies, word id 0 occurs twice in the first document. This will be used as input. By altering the values for Alpha and eta hyperparameters in LDA sparsity of topics could be reduced. Number of documents to analyze could be controlled using the 'Chunksize' parameter followed by 'update_every' to decide Updation frequency of topics. Number of 'passes' specify the total number of training passes required for effective topic extraction.

3.1.3. b. Topic Extraction and Tag Assignment

Topic extraction and tag assignment is the next step after term count extraction. Topics are extracted by traversing different documents that are represented as latent topics, where each topic is characterized by a distribution over words.

Topic extraction for each document q in a corpus c:

1. Choose M Poisson(ξ). 2. Choose P Dir(α). 3. For each of the O words qn:
(a) Choose a topic tn Multinomial(P). (b) Choose a word qn from p(qn |tn,O), a multinomial probability conditioned on the topic qn. In the next step, unigrams, bigrams, trigrams are obtained and lemmatized to populate the dictionary. To improve the efficiency the words appearing in less than 18 documents are removed and only first 100000 very frequent tokens are taken for analysis. And also the tokens were pre tagged with their respective Part of Speech which further added the contextual value to the tokens. The tokens were grouped by comparing the terms with dictionary terms and the corresponding labels were returned as tags. The result obtained after using LDA is given in Fig.3. the numbers at the starting 0,1,2, etc are sample topic tags among the total of 20 tags and values represent the individual term weights in a sample document.

```
[(0,
  '0.053*"uk" + 0.037*"claiming" + 0.026*"hewlett_packard" + 0.024*"cc" + '
  '0.011*"skills" + 0.011*"david_veal" + 0.009*"laughed" + 0.005*"skill" + '
  '0.005*"motto" + 0.004*"cont_education"'),
 (1,
  '0.059*"genocide" + 0.037*"islam" + 0.033*"principles" + 0.030*"moving" + '
  '0.026*"patients" + 0.026*"round" + 0.023*"carl" + 0.022*"hmm" + '
  '0.010*"financial" + 0.009*"benedikt_rosenau"'),
```

Fig. 3 Topics modeled from LDA

Topic modeling produces an intermediate classifier, but due to the presence of more number of features, as each topic was represented by at least 7000 different features, the time taken for classification is increased exponentially, especially when new documents come in large numbers. Secondly, every time a new document has to converted into a thematic map which is also time consuming. Thirdly, possessing a dictionary of words during run time occupies more memory and slows down the classifier performance even if the most effective algorithm is used. To enable scalability and relatively faster classification, we introduced two more layers to the framework. The layers are explained below.

3.2 ONTOLOGY LAYER

It deals with automatic construction of the domain ontology. Once the topics and their relevant terms are obtained from the data layer, it will be parsed into an ontology. This parser will automate the ontology learning process thereby eliminating need for an expert. It has proved to be more effective than extracting ontology from text triples which is prone to errors and also devoid of semantic richness. After parsing all the topics, the domain ontology will be built which will further be converted into a knowledge graph.

3.2.1 Topic Parser

Parser for the news topics files is constructed in python and the parsed dictionaries are saved. In this parser, Regexpr is used to populate topic categories to News ontology. It first extracts the rows related to 20 news categories etc. and then build the ontology on top of this categories. For each category their feature set, that is, the terms related to the topics are parsed in the second level forming the child nodes of the respective topic categories (parent nodes). One part of this program will open the files, filter them, and outputs the filtered files. Another part of the program then converts files to instances of ontology.

3.2.2 Domain Ontology Construction and Graph Database Mapper

These instances become the contents of the ontology and become the wholesome knowledge representation of News domain. The topic parser automated the process of adding information to the ontology which can be accessed using Protégé, and can be converted to different formats other than RDF. We have kept the file format as RDF as it can be mapped to a graph database like neo4j and GraphDB easily. Number of classes obtained from the data corpus is 20 and total number of ontological instances is 1,40,000. Having the data in ontological structure with implicit as well as explicit connections also eliminate the need for multiple table join operations otherwise required in conventional database applications. It also helps to address the problem of dimensionality of text data. It also uncovers other implicit relationships in the topics with the help of reasoner, in this work FacTT++ reasoner was used.

One more interesting aspect of OntoClassifier is it erases the thin line between ontology and graph data[3]. The ontology will be interpreted as a knowledge graph after ontology construction. Since, graph database platforms like neo4j facilitate application of machine learning algorithms on connected data. The entities in knowledge graph are heavily interconnected providing the big picture of any real world object. Ontologies are restricted by reasoners when it comes to text classification also traversing large ontologies is time demanding.

3.3 Classification layer

This is the final and most important part of the proposed architecture. It is concerned with building a semantic data model using state of the art machine learning algorithm and finally evaluates the performance of the OntoClassifier with benchmark algorithms and datasets.

3.3.1 Semantic Data Model

Semantic data models are the need of the hour, given the ambiguity and polysemy of natural language data. The aim of this framework is to construct a text classifier that includes semantic information as well as rich feature space of the data corpus. For achieving this we have improved the Multinomial Naive Bayes Algorithm using Feature Hashing.

Naive bayes is a family of probabilistic algorithms that take advantage of probability theory and bayes' theorem to predict the tag of a text document. they are probabilistic, which means that they calculate the probability of each tag for a given document, and then output the tag with the highest one. bayes' theorem, with probability of features, based on prior knowledge of conditions is constructed. for this ontoclassifier framework the conventional mnb is improved by incorporating hashing trick to enable fast and accurate classifications. since, the data involved is in natural language format a machine learning algorithm efficiency could be improved if search time is reduced in a large feature space.

Improved - Multinomial Naïve Bayes Independence Assumptions

$$P(x1, x2, \ldots, xn \mid c)$$

Conditional Independence: Assume the feature probabilities P(xi|cj) are independent given the class c.

$$P(x1, \ldots, xn \mid c) = P(x1 \mid c) \bullet P(x2 \mid c) \bullet P(x3 \mid c) \bullet \ldots \bullet P(xn \mid c)$$

MULTINOMIAL NAÏVE BAYES CLASSIFIER

$$cMAP = \underset{c \in C}{\operatorname{argmax}}\, P(x1, x2, \ldots, xn \mid c)\, P(c)$$

$$cNB = \underset{}{\operatorname{argmax}}\, P(cj) \prod P(x \mid c)$$

Feature Hashing

It deals with out-of-vocabulary words even a rich feature set like ours can encounter totally new words in documents. It is not necessary to retrain the model from scratch every time we encounter a new word or misspelling, and it is highly accurate without having to keep a huge vocabulary in RAM.

The hash functions, map data of arbitrary sizes to data of a fixed size. Using[23] the dictionary generated by the Term Count Extraction module, a zero column vector with a huge number of elements for each of our training examples is built and a hash function that gets strings and outputs values in the range [0, 2^{28}) is chosen. It ensures that hash function will never address an hash table entry outside the feature vectors' dimensions.

After this, each word is fed one by one, through the hash function, and increment the value at the given hash table by one and derive a sparse matrix. This eliminates the need for having all the features in the form of dictionary in the memory. When a new news document comes, each word is passed through the hash function. Every new word ends up incrementing some feature value. This incrementally fit our model with new examples without retraining the model from scratch.

3.3.2 OntoClassifier

This classifier now has more than 100000 news related features neatly arranged in a knowledge graph with a hash table. The algorithm is also improved with a novel feature hashing technique that interestingly approaches text categorization problem from a information retrieval point of view. Which in a way, eliminated the need for training data and model training time. It uses very low memory and scalable to large datasets as there is no need to store a vocabulary dictionary in memory. After representing domain knowledge as graph it addresses the problem of imbalanced nodes as the number of nodes is fixed in the ontology stage itself. The need for this type of classifier becomes inevitable in situations where data cannot be exhibited in a relational table.

IV. EVALUATION RESULTS

The system is implemented using Jupyter Notebook with Python 3 and all the libraries for the concerned operations were compatible with the python version and Ubuntu 18.04 operating system with 8GB RAM capacity. The carried out experiments aimed to optimize the different parameters of our proposed method and also to compare the effectiveness of our method with benchmark algorithms. Following are the metrics used to describe the efficiency of the model.

```
Classification Error of MultinomialNB: 0.769787573022

Recall = Sensitivity of MultinomialNB: 0.930212426978

Precision of MultinomialNB: 0.928195639892

F-measure of MultinomialNB: 0.934315231835
```

FIG.4 ACCURACY OF THE MULTI-CLASS CLASSIFIER

Accuracy = TP+TN/TP+FP+FN+TN

It is a general notion that higher the accuracy, the better is the model however, it happens only when both the false negatives and false positives are equal in number, which seldom occurs in datasets. Therefore, it is customary to have additional performance measures, such as precision, recall and f1 scores. Initially, our model obtained an accuracy of 88% with a time duration of 5.03 seconds. Which was again improved by tuning the alpha parameter of the multinomial naive bayes algorithm, thereby achieving an accuracy of 93% which is highly efficient as can be seen from figure 4 and table 1.

Precision = TP/TP+FP

It is the ratio of correctly classified positive observations to the total classified positive observations. Higher the precision lower the false positive rate. The proposed model has obtained a precision score of 0.928 as given in figure 4 which is 93%. The improvement is effective when compared to benchmark algorithms given in figures 6, 7 and 8.

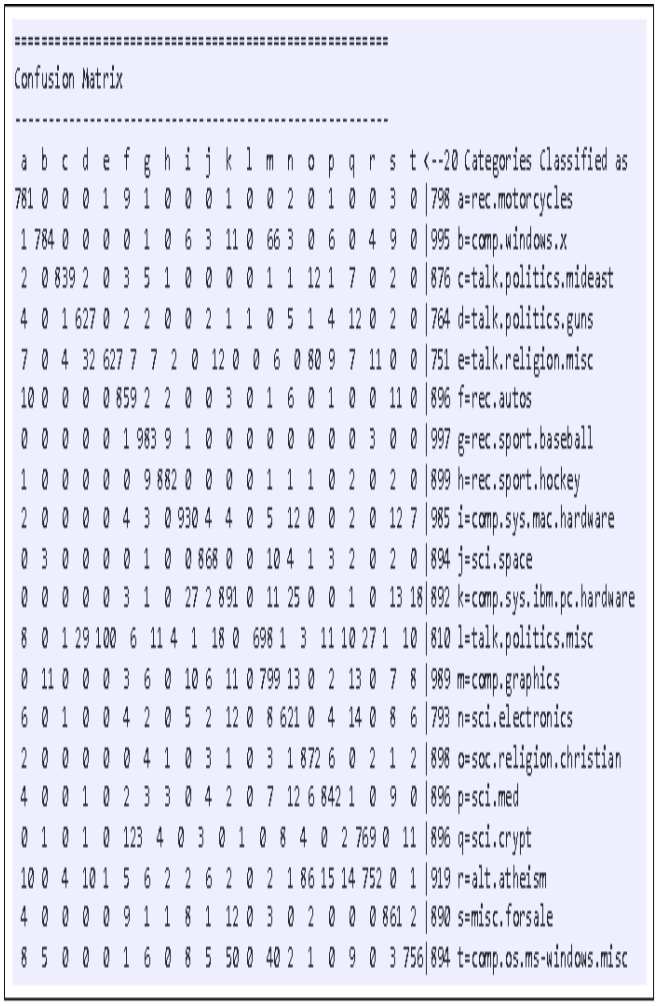
Recall = TP/TP+FN

It is also known as Sensitivity, a ratio between accurately classified positive observations to all the observations obtained in the actual class. The recall value from figure 4 is 0.930 in other words 93% which is good for this model.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

It is the weighted average or measure of optimizing Precision as well as Recall. It depends on both false positives and false negatives. F1 score is valid in cases where the class distribution is highly uneven., The F1 score obtained in our case is 0.934.

The performance of the classifier is plotted as a confusion matrix given in Fig. 5. It depicts the majority of predictions in the diagonal where the predicted label and actual label are equal.



Confusion Matrix

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	<--20	Categories	Classified as
781	0	0	0	1	9	1	0	0	0	1	0	0	2	0	1	0	0	3	0	798	a	rec.motorcycles
1	784	0	0	0	0	1	0	6	3	11	0	66	3	0	6	0	4	9	0	995	b	comp.windows.x
2	0	839	2	0	3	5	1	0	0	0	0	1	1	12	1	7	0	2	0	876	c	talk.politics.mideast
4	0	1	627	0	2	2	0	0	2	1	1	0	5	1	4	12	0	2	0	764	d	talk.politics.guns
7	0	4	32	627	7	7	2	0	12	0	0	6	0	80	9	7	11	0	0	751	e	talk.religion.misc
10	0	0	0	0	859	2	2	0	0	3	0	1	6	0	1	0	0	11	0	896	f	rec.autos
0	0	0	0	0	1	983	9	1	0	0	0	0	0	0	0	0	3	0	0	997	g	rec.sport.baseball
1	0	0	0	0	0	9	882	0	0	0	0	1	1	1	0	2	0	2	0	899	h	rec.sport.hockey
2	0	0	0	0	4	3	0	930	4	4	0	5	12	0	0	2	0	12	7	985	i	comp.sys.mac.hardware
0	3	0	0	0	0	1	0	0	868	0	0	10	4	1	3	2	0	2	0	894	j	sci.space
0	0	0	0	0	3	1	0	27	2	891	0	11	25	0	0	1	0	13	18	892	k	comp.sys.ibm.pc.hardware
8	0	1	29	100	6	11	4	1	18	0	698	1	3	11	10	27	1	10		810	l	talk.politics.misc
0	11	0	0	0	3	6	0	10	6	11	0	799	13	0	2	13	0	7	8	989	m	comp.graphics
6	0	1	0	0	4	2	0	5	2	12	0	8	621	0	4	14	0	8	6	793	n	sci.electronics
2	0	0	0	0	0	4	1	0	3	1	0	3	1	872	6	0	2	1	2	898	o	soc.religion.christian
4	0	0	1	0	2	3	3	0	4	2	0	7	12	6	842	1	0	9	0	896	p	sci.med
0	1	0	1	0	123	4	0	3	0	1	0	8	4	0	2	769	0	11		896	q	sci.crypt
10	0	4	10	1	5	6	2	2	6	2	0	2	1	86	15	14	752	0	1	919	r	alt.atheism
4	0	0	0	0	9	1	1	8	1	12	0	3	0	2	0	0	0	861	2	890	s	misc.forsale
8	5	0	0	0	1	6	0	8	5	50	0	40	2	1	0	9	0	3	756	894	t	comp.os.ms-windows.misc

Fig.5 . Confusion Matrix of OntoClassifier.

As it can be seen from the Fig. 5, it summarizes entire performance of the classifier. From the Fig. 5, it is understood that, class labels talk.religion.misc, talk.politics.guns and sci-electronics have least accurate observations. It is due to more ambiguity in the words or terms involved in such documents. However, the error rate is below acceptable levels in ninety percent of the classes which is the main expectation from a classifier.

The OntoClassifier was tested with major benchmark algorithms to compare the performance. The performance was measured in five different categories such as training time, testing time, accuracy, dimensionality and density. The Table 1 illustrates the different performance metrics for benchmark algorithms used in OntoClassifier. Ridge classifier algorithm performed the worst among all the algorithms and our proposed improved MNB outperformed other algorithms in all aspects of evaluation. Thus, the framework has proved to be robust in differing use cases.

Table 1. Performance of benchmark algorithms in OntoClassifier

Algorithms	Train time	Test time	Accuracy	Dimensionality	Density
Ridge Classifier	0.132s	0.001s	0.796	33809	1.000000
Perceptron	0.017s	0.002s	0.846	33809	0.255302
kNN	0.002s	0.317s	0.858	33809	0.568924
Random Forest	1.671s	0.071s	0.870	33809	0.642156
Imp-Multinomial Naive Bayes	0.003s	0.001s	0.939	33809	1.000000
Linear SVC	0.252s	0.002s	0.780	33809	0.748451

PR - AUC curves

Precision, recall and Area Under Curves(AUC) were also tested and plotted for comparing the accuracy of sample classes of the dataset. It can be seen from figure 6,7 and 8 that PR and AUC of linear SVC algorithm for five classes is obtained, likewise for k-Nearest Neighbor algorithm and the proposed improved-MNB. The class accuracy is greater for the proposed algorithm compared to the other two, which rendered a mediocre AUC value. The greater the AUC value the better is the model accuracy. It is the aggregate performance of possible classification thresholds in other words, classification-threshold-invariant. It is also scale invariant that is concerned only with ranking of classifications rather than a absolute value. From the figures 6,7 and 8 the AUC values are 0.7, 0.5 and 0.8 respectively. Thus, the proposed classifier is evaluated using most prominent performance evaluation metrices and the results prove that the model is highly efficient.

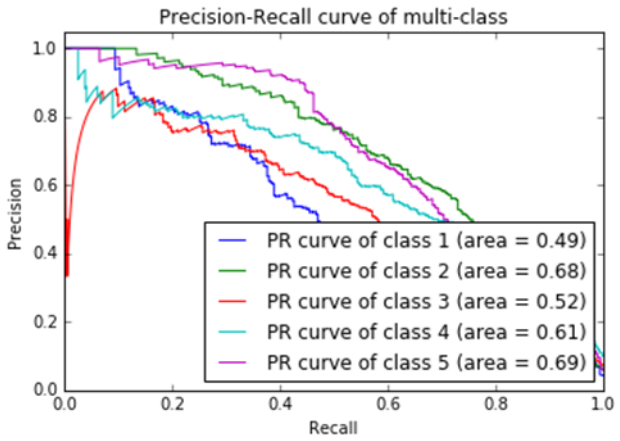


Fig.6. PR and AUC of LinearSVC

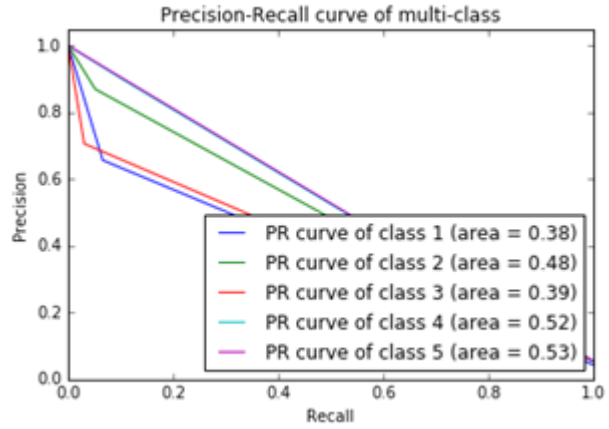


Fig.7. PR and AUC of KNN

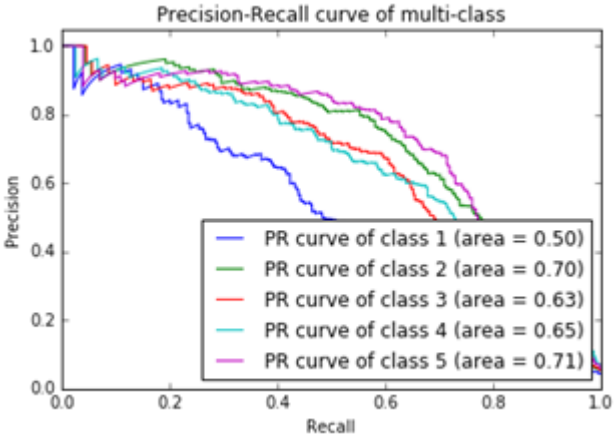


Fig 8. PR and AUC of Improved MultinomialNB

V. CONCLUSION

In this work, we have proposed a novel multi-class classification approach based on ontology derived from natural language texts, which used a training-less classifier without a pre-classified set of training documents. The semantic richness present in natural language texts has been utilized completely to provide accurate classifications. Also, OntoClassifier will reduce the training time required when the number of features exceed the potential of a classification algorithm. As here, the hash table is memorized by the data model and tags are assigned by mapping terms in index and topics in the database. The accuracy is improved approximately 9% by the proposed architecture which is a remarkable improvement compared to baseline models, which is averagely 87.43%. Our experiments, using the 20newsgroup dataset and the crawled data corpus, proved that our method provides a comparable performance to the baseline in terms of AUC and PR curves. In future, more ways to automate ontology construction with additional feature engineering techniques could be investigated..

REFERENCES

1. Berna Altınel, Murat Can Ganiz,(2018)"Semantic text classification: A survey of past and recent advances", *Information Processing and Management54,1129-1153.*
2. NayatSanchez-Pi, LuisMartí, Ana CristinaBicharraGarcia,(2015)" Improving ontology-based text classification: An occupational health and security application", Journal of Applied Logic. http://dx.doi.org/10.1016/j.jal.2015.09.008.
3. ImadZeroual, Abdelhak Lakhouaja,(2018)"Data science in light of natural language processing: An overview", *The First International Conference On Intelligent Computing in Data Sciences , Procedia Computer Science 127 ,82–91.*
4. Berna Altınel, Murat Can Ganiz, (2018)," Semantic text classification: A survey of past and recent advances", Information Processing and Management, 54 ,1129–1153.
5. Abolfazl ravanshad ,(2018), https :// medium. com /@a rava nshad /ensemble-methods-95533944783f_accessed on 24-07-2019
6. Jason D.M. Rennie, Lawrence shih, Jaime Teevan, David R.Karger.(2003).Tackling the Poor Assumptions of Naive Bayes Text Classifiers. *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC.*
7. Mehdi Allahyari, Krys J. Kochut, and Maciej Janik,(2014)" Ontology-based Text Classification into Dynamically Defined Topics", IEEE International Conference on Semantic Computing.
8. Zuo, Zheming, Li, Jie, Yang, Longzhi, Anderson, Philip and Naik, Nitin (2018)"Grooming Detection using Fuzzy-Rough Feature Selection and Text Classification", In:FUZZIEEE 2018 - IEEE International Conference on Fuzzy Systems, 8th - 13th july 2018, Rio deJaneiro, Brazil.
9. Gaurav Singh,James Thomas,Iain J. Marshall,John Shawe-Taylor,Byron C. Wallace,(2018)"Structured Multi-Label Biomedical Text Tagging via Attentive Neural Tree Decoding",*arXiv:1810.01468v1 [cs.IR] oct 2.*
10. NayatSanchez-Pi, LuisMartí, Ana CristinaBicharraGarcia, (2015)" Improving ontology-based text classification: An occupational health and security application",Journal of Applied Logic, http://dx.doi.org/10.1016/j.jal.2015.09.008.
11. Roger Alan Stein ,Patricia A. Jaques , João Francisco Valiati (2019) " An analysis of hierarchical text classification using word embeddings", *Information Sciences 471 ,216–232.*
12. Krzysztof Wr´obel, Maciej Wielgosz, Aleksander Smywi´nski-Pohl, and Marcin Pietron,(2016)" Comparison of SVM and Ontology-Based Text Classification Methods", Springer International Publishing Switzerland, ICAISC 2016, pp. 667–680,.DOI: 10.1007/978-3-319-39378-0 57.
13. Berna Altınel, Murat Can Ganiz, (2018)," Semantic text classification: A survey of past and recent advances", Information Processing and Management, 54 ,1129–1153.
14. Chumsak Sibunruang and Jantima Polpinij,(2014)" Ontology-Based Text Classification for Filtering Cholangiocarcinoma Documents from PubMed", Springer International Publishing Switzerland, pp. 266–277
15. M. Thangaraj, V. Gayathri, (2013) "A Context-Based Technique Using Tag-Tree For An Effective Retrieval From A Digital Literature Collection", Journal of Computer Science, Vol.9, no.11, , pp 1602-1617.
16. Lin Liu, Lin Tang, Wen Dong, Shaowen Yao and Wei Zhou,(*2016*)," An overview of topicmodeling and its current applications in bioinformatics", Liu *et al. SpringerPlus 5:1608*,DOI 10.1186/s40064-016-3252-8.
17. Wael Alkhatib, Saba Sabrin, Svenja Neitzel, and Christoph Rensing,(2018)" Towards Ontology-Based Training-Less Multi-label Text Classification", Springer International Publishing Switzerland AG, NLDB, LNCS 10859, pp. 389–396, 2018. https://doi.org/10.1007/978-3-319-91947-8_40.
18. M. Thangaraj**,** M.Sivakami, (2018)" Text Classification Techniques: A Literature Review", Interdisciplinary journal of information knowledge and management , Informing Science Institute Volume 13, 117-135.
19. Jose Aguilar , Marxjhony Jerez, Taniana Rodríguez, (2018)" CAMeOnto: Context awareness meta ontology modeling", Applied Computing and Informatics 14 ,202–213.
20. M . Thangaraj**,** G Sujatha, (2014)"An architectural design for effective information retrieval in semantic web" International journal of Expert systems with Applications, Vol 41, Elsevier.
21. Mohamed K. Elhadad, Khaled *M.* Badran, ,Gouda I. Salama,(2017),"A Novel Approach for ontology-based Dimensionality Reduction for Web Text Document Classification" , *IEEE, May 24-26.*
22. Dr.M.Thangaraj**,** Mrs.M.Chamundeeswari,(2013)"Agent Based Personalized Semantic Web Information Retrieval System", International journal of Advanced Computer Science and Applications, vol 5, no 8.
23. Roberto De Virgilio, Antonio Maccioni, and Riccardo Torlone,(2014)" Model-Driven Design of Graph Databases", Springer International Publishing Switzerland, pp. 172–185.

AUTHORS PROFILE

Dr.M Thangaraj, Department of Computer Science, School of Infor- mation Technology, Madurai Kamaraj University, India.

M Thangaraj received his post-graduate degree in Computer Science from Alagappa University, Karaikudi, M.Tech. degree in Computer Science from Pondicherry University and Ph.D. degree in Computer Science from Madurai Kamaraj University, Madurai, TN, South India in 2006. He is now the PROFESSOR and Head of Computer Science Department at M.K.University. He is an active researcher in Big Data Analytics, Social Media Analytics ,Wireless Sensor Networks and has published more than 130 papers in Journals and Conference Proceeding.

Ms. M Sivakami, Department of Computer Science, School of Information Technology, Madurai Kamaraj University, India.

M Sivakami received her post-graduate degree in Computer Science in 2012, from Annamalai University, Chidambaram and M.phil. degree in Computer Science from Madurai Kamaraj University in 2015. She is now a Research Scholar of Computer Science in the Department of Computer science at M.K.University. She is an active researcher in Text Analytics.