Semantic Ontology-Based Approach to Enhance Text Classification

Sonika Malik[a,b], Sarika Jain[a]

[a]*National Institute of Technology, Kurukshetra*
[b]*Maharaja Surajmal Institute of Technology*
sonika.malik@gmail.com, jasarika@nitkkr.ac.in

Abstract

Text Classification is the process of defining a collection of pre-defined classes to free-text. It has been one of the most researched areas in machine learning with various applications such as sentiment analysis, topic labeling, language detection and spam filter etc. The efficiency of text classification improves, when some relation or pattern in the data is given or known, which can be provided by ontology. It further helps in reducing the size of dataset. Ontology is a collection of data items that helps in storing and representing data in a way that preserves the patterns in it and its semantic relationship with each other. We have attempted to verify the improvement provided by the use of ontology in classification algorithms. The code prepared in this research and the method developed is pretty generic, and could be extended to any ontology based text classification system. In this paper, we present an enhanced architecture that can uses ontology to provide an effective text classification mechanism. We have introduced an ontology based text classification algorithm by utilizing the rich semantic information in Disease ontology (DOID). We summarize the existing work and finally advocate that the ontology based text classification strategy is better as compared to conventional text classification in terms of different metrics like Accuracy, Precision, Recall, and F-measure etc.

Keywords

Text Classification, Ontology, Semantic AI, Symbolic AI, Statistical AI, Classifier.

1. Introduction

The classification of entities based on the available data is the foundation for classification techniques. The available data could be of two types- the information that we have on hand, and the information that we have previously used for classification. Either way, an accurate and precise classification relies on the amount of information that is available to us. The ways of processing and analysing information has been transformed through digitization.

There is a plethora of textual data everywhere we look around, from magazines to journals to papers. There is a need to systematically categorize and interpret this information without compromising time. Automated text classification [1] is one of the most helpful tools for this.

It's one of the most important and rudimentary features in Natural Language Processing (NLP) [2], with broad applications such as sentiment analysis [3], topic labelling, spam detection [4], and intent detection [5]. Text classifier [6] are made and meant to be implemented on a diverse range of textual datasets. Text classification can work on both, structured and unstructured datasets. To understand the process of classification and how ontology fits in this process, there is a hierarchical progression as shown in Figure 1.

Artificial Intelligence (AI) is anticipated to produce hundreds of billions of dollars in economic value. However, considering that technology forms part of our

ISIC'21: International Semantic Intelligence Conference, February 25-27, 2021, Delhi, India
✉:sonika.malik@gmail.com (S. Malik); jasarika@nitkkr.ac.in (S. Jain)
: 0000-0003-2721-1951(S Malik)
© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

everyday lives, many people remain suspicious. Their key issue is that AI approaches perform like black-boxes and seems to generate ideas without any explanation. In addition, many industries recognised knowledge graphs (KGs) as an effective method for data processing, management and enrichment. Although KGs are also increasingly recognisable as a foundations of an AI system that makes explainable AI via the design concept called "Human-in- the-Loop" (HITL). The AI's promise is to automatically derive patterns and rules from massive datasets based on machine learning algorithms such as deep learning. This fits very well with particular issues and helps to simplify classification activities in many situations. The machine learning algorithms gain the knowledge from historical information, but they cannot derive new results from it. Without explanation, there is no confidence. Explain ability ensures that trustworthy agents in the system are able to understand and justify the AI agent's decisions [50].

Semantic AI integrates symbolic AI and statistical AI. It incorporates the approaches like machine learning, information analysis, semantic web and text mining. It combines the benefits of AI techniques, primarily neural networks and semantic reasoning. It is an improvement of the existing framework used primarily to create AI-based systems. This brings fast learning from less trained data, for example chatbots can be developed without cold-start problem. Semantic AI incorporates a radically different approach and therefore complementary skills for additional stakeholders. Although conventional Machine Learning is primarily performed by data or information scientists involved in Explainable AI or semantic AI. At the heart of Semantic Enriched Artificial Intelligence architecture, a semantic knowledge graph is used by providing the means for a more automated data quality management [7]. For the better quality data and more options in feature extraction, semantically enhanced data works as a base. It gives the better accuracy of classification and prediction intended by machine learning algorithms. Semantic AI aims to have an infrastructure to address the knowledge asymmetries between designers of AI applications and other stakeholders including customers and decision makers, in direct reference to AI systems which 'work like magic' where only some of the analysts actually recognise the fundamental techniques [8].

Ontology- Ontology specifies a conceptualization of a domain in terms of concepts, attributes, and relations [49]. In simple terms, Ontology is analogous to a dictionary, which stores the information about entities. This information usually consists of the features and relations of the said entities. The immense importance of ontology is utilised in the research fields, such as data science, where, it eases information processing because of its organised structure, as compared to the more conventional ways of processing raw data. The formal ontology, thus, represents data in an organised way and used as a framework [9].

Ontology based text Classification- For Machine Learning (ML) style classification, algorithms such as Naive Bayes (NB) [10] or Support Vector Machine (SVM) [11] etc. are used, where we train a model to read text as feature vectors and output as one of *n* classes. One use of ontology would be to mark-up entities in the text. In our case, we have a medical ontology like DOID, whose nodes have information about various diseases, symptoms, medications, etc.

We could look for these entities in our text and mark them as single entity - so for example, if we found the string "Lung Cancer" in our text which is also a node in our ontology, we could replace all occurrences of "Lung Cancer" with a single token "Lung_Cancer" and treat this token as a feature for our classification. These ontology nodes usually contain multiple versions of the string that represents it. For example, "heart attack" is also known as "myocardial infarction", so if our text contains either string, they could be normalized down to one single string and treated as a single feature for classification. For rule-based classifiers such as Bayesian Networks or decision tree algorithms [12], we could also leverage the knowledge in the ontology to create generalized rules.

The remaining paper has been organised as follows: Section 2 describes the related work in the field of text classification. Section 3 defines the background knowledge. Section 4 presents the assessment of proposed system. Section 5 describes the comparison and results and finally paper ends with conclusion and future scope.

2. Related Work

Angelo A. Salatino, Thiviyan Thanapalasingam, Andrea Mannocci, Francesco Osborne and Enrico Motta [13] came up with the Computer Science Ontology (CSO). The CSO consists up to twenty-six thousand domains, and as many as two hundred and twenty-six thousand interpretable relations between

these domains. To support its availability, they also developed the CSO Portal, a web application which allows users to explore the ontologies and send feedback.

Angelo A. Salatino, Francesco Osborne and Enrico Motta [14] introduced the CSO Classifier for automatic classification of research papers according to the Computer Science Ontology (CSO). It is an unsupervised approach. For every research Meta data, the CSO takes as input, it returns a list of suitable topics that could be used in classifying the said input research paper.

Angelo A. Salatino, Francesco Osborne and Enrico Motta [15] presented a CSO classifier for automatic classification of academic papers according to CSO's rich taxonomy of subjects. The aim is to promote the acceptance of CSOs through the various communities involved in scholarly data and enable the creation of new applications that rely on this knowledge base. This paper proposed four stages:

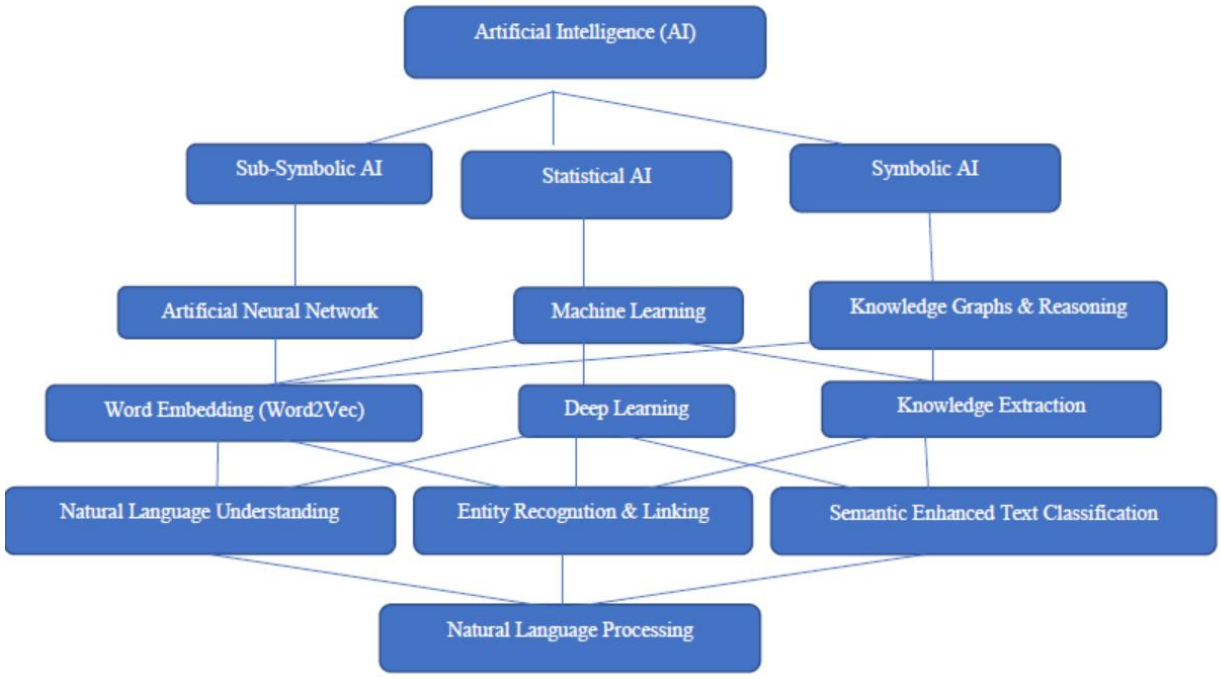

Figure 1. Concept Hierarchy in Semantic AI [45, 46]

(a) Constructing research ontology, (b) Classifying new research proposals into disciplines, (c) building research proposal clusters using text mining, (d) balancing research proposals and regrouping them by considering applicants' characteristics.

Preet Kaur and Richa Sapra [17] also researched in a similar domain, wherein, they proposed Ontology-Based text mining methods for classification of research proposals as well as external research reviewers.

Chaaminda Manjula Wijewickrema and Ruwan Gamage [18] addressed the fallacies in manual classification and proposed ontology based methods for fully automatic text classification.

A Sudha Ramkumar, B Poorna and B. Saleena [19] used WordNet ontology to perform ontology based clustering of sports related terms, so as to preserve the semantic meaning behind terms while clustering them.

Nayat Sanchez-Pi, Luis Marti and A.C.B. Garcia [20] presented a probing algorithm for the automatic detection of accidents in occupational health control. The proposal has more accurate heuristics because it contrasts the relevance of techniques used with the terms. The basic accident detection problem is divided into three parts: (i) text analysis, (ii) recognition and (iii) classification of failed techniques which caused accidents.

Decker [21] presented a different approach to categorize research papers by using the words present in the papers abstract. It is an unsupervised method which evaluates the relevance of suitable topics for the research paper on various time scales.

Herrera et al. [22] devised a way to categorize research papers specific to the domain of physics. They did this with the help of PACS, which stands for Physics and Astronomy Classification Scheme. They created a network like structure where, a PACS code was assigned to every topic node, and a connection between two nodes was possible only if their codes co-occur together in at least one paper.

Ohniwa et al. [23] gave a similar analysis in the field of biomedicine. They used the Medical Subject Heading (MeSH).

Mai et al [24] showed that the performance of their model, which was only trained using titles, was as good as the models trained by mining the full texts of papers and articles. They developed their approach using deep learning techniques. As training set, they used scientific papers from EconBiz and PubMed, respectively annotated with the STW Thesaurus for Economics (approximately five thousand classes) and MeSH (approximately twenty-seven thousand classes).

Cook et al. [25] developed a method of allocation of papers to reviewers optimally, to aid the selection process.

Arya and Mittendorf [26] suggested a rotation based method for the assignment of projects.

Choi and Park [27] offered a solution for Research and Development proposal classification, which was text mining based.

Girotra [28] proposed a study for the evaluation of portfolio projects.

Sun et al. [29, 30] developed a mechanism for assessment of reviewers, who would evaluate the research papers. Mehdi Allahyari, Krys J. Kochut and Maciej Janik [31] proposed a way of dynamic classification of textual records in dynamically generated classes.

Rudy Prabowo, Mike Jackson, Peter Burden and Heinz-Dieter Knoell [32] developed a web page classifier. Its classification was with reference to the Dewey Decimal System and Library of Congress Classification schemes.

3. Background Knowledge

In this section we have discussed the pre-processing steps for textual data and Machine Learning classifiers that are being used in our research.

3.1. Pre-Processing Textual data

According to the official documentation, *the Natural Language Toolkit (NLTK) [48] is a platform used for building Python programs that work with human language data for applying in statistical Natural Language Processing (NLP)*. It is a useful tool in python, which helps in processing a diverse range of languages by providing algorithms for it. This tool is powerful because it is free and open source. Also, one does not need to look for any special tutorials when using the NLTK, as its official documentation is very well described. The most common algorithms used in NLTK are tokenization, lemmatization, part of speech tagging etc. These algorithms are essentially used to preprocess textual data. The preprocessing takes place in five parts:

Tokenization- A token is the fundamental building block of any linguistic structure, such as a sentence or a paragraph. The process of tokenization is to break these structures down into tokens. Tokenizer could be of two types – a sentence tokenizer's tokens are sentences. It, therefore, breaks paragraphs down into sentences. A work tokenizer identifies words as tokens. It, hence, disintegrates sentences into words.

Stemming- A stem is the root word or phrase from which different forms of that word could be derived. Stemming is the process of identifying all the words that were derived from the same stem and reduce them or normalize them back to their stem form. For example, connection, connected, connecting word reduce to a common word "connect".

Lemmatization- Sometimes, we may encounter words that have different stems but the same final meaning. In such a case, there is a need for a dictionary lookup to further reduce the stems to their common meaning, or the base word. This base word is known as lemma, and hence the name lemmatization. For example, the word "better" has "good" as its lemma. Such cases are missed out during stemming because these two words not at all alike, and would need a dictionary lookup where, their meanings can confirm the lemma.

POS Tagging- It stands for Part of Speech, and just as the name suggests, it identifies the various parts of a linguistic structure like a sentence. The different parts could be an adjective or a noun or a verb. It does so by studying the structure of the sentences and observing the arrangement of words and the relation between the various words.

3.2. Text classification and classifiers

The idea behind text classification is to group text into categories using machine learning. It finds use in many relevant areas such as sentiment analysis, emotion analysis, etc. There have been many classifiers developed for each classification category. As stated previously, text classifier is made and meant to be implemented on a diverse range of textual datasets. Text classification can work on both, structured and unstructured datasets. Both types of datasets find numerous applications in various fields. The Classification process in machine learning can be explained very simply. First, we assess and analyze the training dataset for boundary condition purposes. Next, we predict the class for new data using the information obtained and learned during the training phase. This is essentially the whole process of classification. Classification could be either supervised or unsupervised. Supervised classification [33] of works on the principle of training and testing, and uses labeled data, i.e. predefined classes, for prediction. In the training phase, the model is made to learn some predefined classes by feeding it labeled or tagged data. In the testing phase, the efficiency of the model's prediction or classification is measured by feeding it unobserved data. In other words, it can only predict those classes in the testing phase which, it has learnt in the training phase. Some common examples of supervised classification are spam filters, intent detection, etc. Unsupervised classification [34, 35] involves classification by the model without being fed the external information. In this, the algorithm of the model tries to group or cluster data points based on similar traits, patterns and other common features that can be used to tie two data points together. A common example where unsupervised classification is really helpful is the search engines. They create data clusters based on insights generated from previous searches. This type of classification is extremely customizable and dynamic as there is no need for training and tagging for it to work on textual datasets. Thus, the unsupervised classification is language compatible. The classifiers used for text-classification could be ML based, such as Naïve-Bayes Classifier, Decision Tree Classifier [36] etc., or it can be based on Neural Network architecture such as Artificial Neural Network, Convolutional Neural Network etc. [37].

The machine learning based classifiers that can be used for text classification are:

(a) Naive Bayes classifier [38, 39] - It uses the Bayes theorem to predict values. This algorithm is good for multi-class classification. Consider a data point x, in a multi-class scenario with three classes- A, B and C. Using Naïve Bayes, we try to predict whether the data point x belongs to class A or B or C, by calculating its probability for the three classes as given in Eq. 1.

$$P(A|B) = (P(B|A).P(A))/P(B) \quad (1)$$

This algorithm is called 'Naïve' because it assumes that all the features are independent of each other as defined in Eq. 2.

$$P(f_1, f_2, f_3 \dots f_n) = P(f_1) = P(f_2) = \cdots P(f_n) \quad (2)$$

There are further two categories of the NB Classifier one is Gaussian NB Classifier and other one is multinomial NB Classifier.

The Gaussian Naïve-Bayes classifier is used when a dataset has continuous values of data. It uses the Gaussian Probability Distribution function (values are centered on mean and as the graph grows, the values decrease). The Multinomial Naive Bayes algorithm assumes the independence of features, and the multinomial component of this classifier ensures that the distribution is multinomial in its features.

(b) Decision Tree [38] - It is a highly intuitive algorithm which uses greedy approach. To construct a decision tree, we have to perform the following steps – (1) select a feature to split the data, (2) select a method to split the data on the said feature. It has the internal working algorithm as: *(i) Create/Select a node. (ii) If the node is pure, output the only class. (iii) If no feature is left to split upon, and the node is impure, output the majority class. (iv) Else find the best feature to split upon. Recursively call on this split. Go to b.*

(c) K-Nearest Neighbor [38, 40] - Consider a scenario where we have to predict to which class, the testing point belongs to, by considering all the features at once. Such is the working of KNN algorithm as shown in Figure 2. To predict the class of the testing data point, we check its vicinity. To classify the testing point, we check a specific number of points (1, 3, 5, 7, etc.) and whichever class is in majority among those, that one is predicted. To select the nearest point, we have to consider its distance from the other points. The distance metric can be (a) Manhattan distance, (b) Euclidian Distance, (c) Minkowski distance.

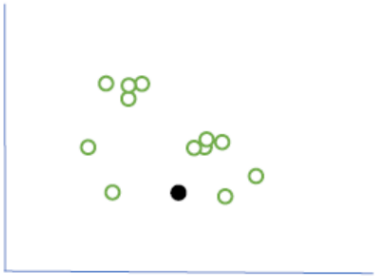

Figure 2. KNN

(d) Random forest [38, 41] - It is an extension of the decision tree classifier. This algorithm uses multiple combinations of decision trees to accurately predict testing data. The random forest classifier overcomes the over-fitting problem of decision trees by building multiple decision trees and going with the majority result. The trees' outputs vary because each tree is built with random data and random features. To generate randomness in trees, we use two techniques-

(i) Bagging: If we have 'm' data points, we select a subset of 'k' out of them. For 'n' trees, n*k subsets are selected. Data points can be considered with replacement as the selection is random; therefore, these trees are called bag trees.

(ii) Feature Selection: In the training phase, some features are selected at random in this technique, with the condition that the selection is performed without replacement.

(e) SVM Classifier [38, 42] - It is a very powerful algorithm and overcomes the limitation of logistic regression. As logistic regression uses sigmoid function, the value predicted for a testing data point is close to 0.5. This causes the problem of incorrect prediction. So, SVM uses the rules of logistic regression only, but exponentially increases the value, so that the values predicted do not fall in the range (-1, 1).

This cost function changes to the following equation in SVM as given in Eq.2.

$$(\theta) = C\sum[\, y^{(i)}\, cost_1\left(\theta^T x^{(i)}\right) + \left(1 - y^{(i)} cost_0\left(\theta^T\, x^{(i)}\right)\right] + 0.5\sum(\theta_i)^2 \quad (3)$$

(f) Logistic Regression [38]: It is a primitive classification algorithm which uses the sigmoid function as in Eq. 4 at its core to perform classification.

$$F(x) = \frac{1}{1 + e^{-x}} \quad (4)$$

As the sigmoid has an exponential function, the graph moves exponentially either towards 0 or 1 with a slight change in x.

The cost function of the binary logistic regression is given in Eq. 5.

$$E(h(x)) = \sum(-y_i \log\left(h(x)\right) - (1 - y_i) \log\ (1 - h(x))) \quad (5)$$

(g) Bagging Classifier [43]: A Bagging Classifier is an ensemble Meta estimating system that fits base classifiers in each of the random subsets of the original data sets and then combines their individual predictions to form a final prediction. Usually, such a meta-estimator can be used to minimize the variance of a black-box estimator by randomization.

4. Proposed Study

The classification by Machine Learning algorithms is supposed to improve with the use of ontology. We aim to verify this fact by studying and comparing values of metrics such as accuracy, precision, recall and F1 score for ontology based text classification and conventional text classification.

4.1. Conventional Text Classification

In the conventional classification the framework had three main phases, (i) Dataset generation (ii) Model training and testing (iii) Analyzing/Classifying results as shown in Figure 3.

1. Dataset Generation: A premature knowledge database of disease-symptom associations was available on [45] which consist of three columns named as disease name, count of disease occurrence and the symptoms; however, it needed modification to be used for our research. Also some new information was added to the dataset so that matching could be done precisely. Thus the final dataset created, is the one that was used for this proposed research. The modified dataset and the ontology are compatible as they consist of classification/output feature "disease name" and the matching feature "disease description". After the ontology and a working dataset were obtained, cleaning and preprocessing of the dataset was done, NLTK is used for processing the dataset. A synthetic dataset is also generated which involves creating new data using programming techniques. In this research we created multiple entries using the random feature value selection of same class. For example, consider a disease having 10 symptoms. We randomly select a subset of these 10 symptoms and generate a new entry for the

dataset involving fewer symptoms and the disease name. This process helps to bind the symptom values to the disease and generate strong positive relation between feature values (symptoms) and class (disease).

2. 2. Model Training/Testing: This phase involves using dataset and applying machine learning classifiers to it. As the dataset initially contains text keywords, it needs to be converted into numbers using count vectorizer module. After this the training data is ready for feeding to the classifier for training. The classifiers used are KNN, SVM, Logistic Regression, Decision Tree and Random Forest etc. After training we can use the model for predictions on testing data. The ratio of training and testing data is 80 and 20 respectively.
3. 3. Analyzing/ Classifying Results: To analyze the results, we compare the disease predictions for the testing data with the actual disease class. After comparing we calculate the classification metrics like accuracy, precision, recall, F1-score. After this computation we can compare the performance of multiple classifiers based on metrics. Also we can verify which classes seem to perform well base on individual class-wise precision and recall values.

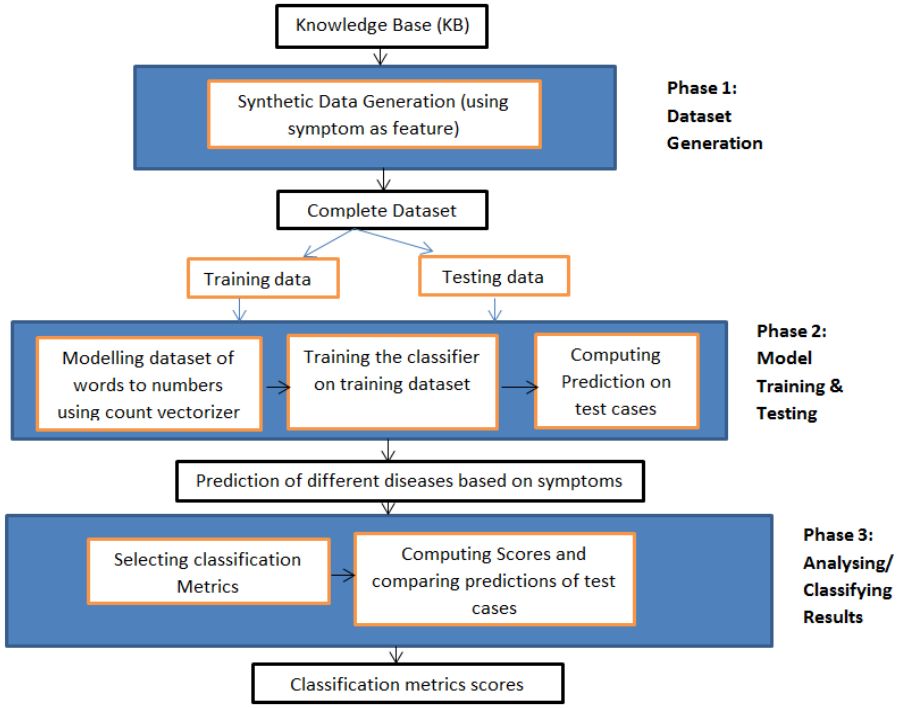

Figure 3. Conventional Text Classification

4.2. Ontology Based Text Classification

For the purpose of this research, we have used the Human Disease Ontology, which was hosted at the website for the Institute of Genome Sciences, Maryland School of Medicine [44]. This ontology is comprehensive hierarchical controlled vocabulary for human disease representation. It consists of unique label for each disease which acts as identifier. The owl file of the ontology was exported to csv file using Protégé. We have presented a second phase between dataset generation and Model training/Testing, in which a hybrid approach for text classification is used to optimize it. The presented methods/phases are: (i)Dataset generation, (ii) Ontology Matching (iii) Model Training and Testing iv) Analyzing and classifying results.

The phases i, iii and iv are explained earlier in section 4.1.

Ontology Matching**:** In this phase the keywords formed from the description of the disease are matched with the keywords of ontology nodes. All the matched nodes are possible classes which can be

used to create the subset of the data for efficient model training. The use of priority based matching helps us to further limit classes. In our research for ontology matching each keyword is assigned two numbers to specify its priority. The first number describes frequency of the keyword and second number describes whether the keyword can be lemmatized or not. If it cannot be lemmatized it is assigned as 1 otherwise 0. Thus each keyword has syntax (name, first priority number, second priority number). The steps for ontology matching are given in Figure 4.

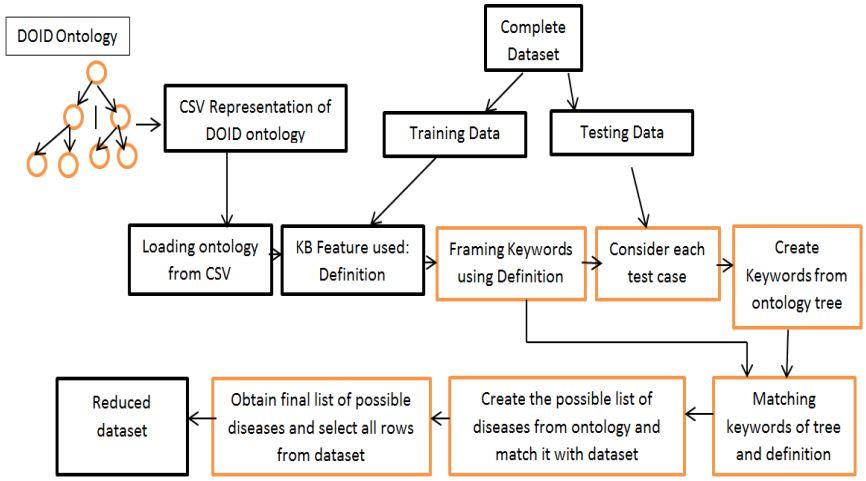

Figure 4. Ontology Matching

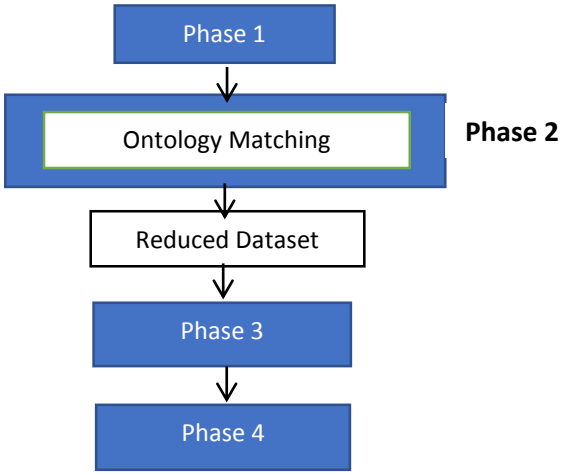

Figure 5. Ontology Based Text Classification

Algorithm 1: Ontology Based Text Classification

The Ontology matching function used in this Algorithm refers to Algorithm 4.2

DOID: Disease Ontology

data_x, data_t= synthetic_data_generation (Knowledge_base) *// Phase1*

x_train, x_test,y_train, y_test =train_test_split (data_x, data_y)

```
        Ontology_tree= Loading_Ontology ()                                  //Phase 2
        for i in range (1, len (Knowledge_base))
        Keywords_for_matching =Keywords_formation (Knowledge_base)
        all_classes= set (data_y)
for z in range (0, len (x_test))
        keywords=keywords_selection (Keywords_for_matching, y_test[z])
        possible_classes=Ontology_matching (tree, keywords)
        for i in range (0, len (possible_classses))
        for j in range (0, len (all_classes))
        if set (word_tokenize (possible _classes[i])). subset (set (word_tokenize (all_classes[j])))
        final_classes.append(all_classes[j])
        else
        continue
        for i in range (0, len(y_train))
                if y_train[i] in final _classes
        indices_to_use.append[i]
                else
                continue
        reduced_x_train, reduced_y_train = reducing_dataset (x_train,y_train, indices_to_use)

training_data = count_vectorizer.fit_transform(reduced. x_train)                //Phase 3
testing_data = count_vectorizer.transform (x_test[z])
classifer.fit (training_data, reduced_y_train)
prediction = classifier.predict(testing_data)
predictions. append (prediction)
accuracy_score = accuracy (predictions, y_test)                                 //Phase 4
precision_score = precision (predictions, y_test)
recall_score = recall (predictions, y_test)
F1-score = F1-score (predictions, y_test)
```

Algorithm 2: Ontology Matching

```
def ontology_mathing (tree, keywords):
    nodes_to_search= []
  found_nodes = []
  priority_1 = []
  priority_2 = []
  priority_3 = []
  for i in range (0, len (keywords)):
    if keywords[i][1]! = 1:
      priority_1. append (keywords[i][0])
    else:
      if keywords[i][2] == 1:
        priority_2. append (keywords[i][0])
      else:
        priority_3. append (keywords[i][0])
  priority_1_count = 0
  priority_2_count = 0
```

```
priority_3_count = 0
for i in range (0, len(tree)):
    priority_1_count = 0
    priority_2_count = 0
    priority_3_count = 0
    for j in range (0, len(priority_1)):
        if priority_1[j]. lower () in [x.lower() for x in tree[i].keywords]:
            priority_1_count=priority_1_count+1
        else:
            continue
    for j in range (0, len(priority_2)):
        if priority_2[j]. lower () in [x. lower () for x in tree[i]. keywords]:
            priority_2_count = priority_2_count + 1
        else:
            continue
    for j in range (0, len (priority_3)):
        if priority_3[j]. lower () in [x. lower () for x in tree[i]. keywords]:
            priority_3_count = priority_3_count + 1
        else:
            continue
    if priority_1_count+priority_2_count+ priority_3_count>=1:
                found_nodes. append ((tree[i]. entity))
    else:
        continue
    classes= []
for i in range (0, len(found_nodes)):
    classes.append(found_nodes[i])
classes=list(set(classes))
# print(classes)
return classes
```

5. Comparison & Analysis of Results for Classification of Various Algorithms with and without Ontology

Parameters used for evaluation and comparison of the model when used with ontology v/s when used without ontology is accuracy, precision, recall, and F1 score.

Accuracy describes the intuitiveness achieved by a model after training. It takes into account all the correctly predicted observations from the list of all predictions.

Accuracy= Number of correct predictions/ total number of Predictions

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

Precision- Sometimes, a classifier may label a class as true for classification of some raw data, when in fact, it should have been false. This is the case of a false positive. Precision takes into account the false positives as well.

$$\text{Precision} = \frac{TP}{TP+FP}$$

Recall- When a classifier marks a class as negative for an unobserved data item, when in fact, it should've been true, it is a case of a false negative. Recall accounts for the sensitivity of a model by taking into account the false negatives.

$$\text{Recall} = \frac{TP}{TP+FN}$$

F1 Score is the weighted average of Precision and Recall. Therefore, it factors in false positives as well as false negatives. In case of an uneven class distribution, F1 score becomes more important than accuracy. Other times, when false negatives and positives have the same cost, accuracy may be treated as the superior evaluation parameter.

$$\text{F1 Score} = \frac{2*(Recall*Precision)}{(Recall+Precision)}$$

Where TP- True Positive
TN- True Negative
FP- False Positive
FN- False Negative

Table 1

Comparison Table for different classifiers based on ontology and without ontology 100 & 500 Test Cases

Classifier	Parameter	100 Test Cases		500 Test Cases	
		Without Ontology	With Ontology	Without Ontology	With Ontology
Naïve-Bayes	Accuracy	0.98	1.0	0.97	0.978
	Precision	0.98	1.0	0.97	0.978
	Recall	0.98	1.0	0.97	0.978
	F1 Score	0.98	1.0	0.97	0.978
Decision Tree	Accuracy	0.74	0.80	0.758	0.852
	Precision	0.74	0.80	0.758	0.852
	Recall	0.74	0.80	0.758	0.852
	F1 Score	0.74	0.80	0.758	0.852
KNN	Accuracy	0.81	0.86	0.818	0.854
	Precision	0.81	0.86	0.818	0.854
	Recall	0.81	0.86	0.818	0.854
	F1 Score	0.81	0.86	0.818	0.854
Random Forest	Accuracy	0.96	0.97	0.932	0.952
	Precision	0.96	0.97	0.932	0.952
	Recall	0.96	0.97	0.932	0.952
	F1 Score	0.96	0.97	0.932	0.952
SVM	Accuracy	0.98	1.0	0.986	0.992
	Precision	0.98	1.0	0.986	0.992
	Recall	0.98	1.0	0.986	0.992
	F1 Score	0.98	1.0	0.986	0.992
Bagging	Accuracy	0.87	0.89	0.894	0.912
	Precision	0.87	0.89	0.894	0.912
	Recall	0.87	0.89	0.894	0.912
	F1 Score	0.87	0.89	0.894	0.912
Logistic Regression	Accuracy	0.99	1.0	0.982	0.99
	Precision	0.99	1.0	0.982	0.99
	Recall	0.99	1.0	0.982	0.99
	F1 Score	0.99	1.0	0.982	0.99

In reference to the above Table 1, the values of the metrics Accuracy, Precision recall and F1-score have the same magnitude. This is due to the fact that the FP & FN values are same in magnitude as there is less number of records per disease. This results in an equal value of precision, recall, F1-score for each class as shown in Figure 6.

It can also be observed that decision tree classifier shows the highest boost in accuracy, precision, recall and F1 score being 0.75 for simple classification, and 0.85 for ontology based classification. There is a 10% improvement in metrics for 500 test cases and 6% for 100 test cases for decision tree classifier. It is followed by the KNN Classifier, which shows the 5% improvement for both 100 & 500 test cases. Rest other classifiers have shown improvement in metrics of around 1% - 3% as shown in Table 1.

The order of classifier w.r.t its magnitude is as follows:

The bagging classifier follows next with the values of the parameters being 0.87 each in simple text classification, and 0.89 in ontology based classification. Random Forest Classifier comes next, as it shows the values of the parameters as 0.96 and 0.97 in each classification case. Naïve-Bayes Classifier and the SVM classifier have the same values of all the parameters as 0.98 for simple classification and 1.0 for ontology based text classification. Now we will come to Logistic Regression, which can be labelled as best classifier with the values of metrics as 0.99 for simple Text classification while 1.0 for ontology based text classification. There is minute difference in the value of metrics of these classifiers. Also the order of the increasing accuracy of various classifiers (with or without using ontology) goes on as:

Decision Tree Classifier < KNN Classifier < Bagging Classifier < Random Forest Classifier < Naïve Bayes Classifier < SVM classifier< Logistic Regression.

	precision	recall	f1-score	support
Alcohol Dependence	1.00	1.00	1.00	1
influenza	1.00	1.00	1.00	3
neoplasm	1.00	1.00	1.00	3
hernia	1.00	1.00	1.00	3
fibrous tumor	1.00	1.00	1.00	1
osteomyelitis	1.00	1.00	1.00	1
pancreatitis	1.00	1.00	1.00	1
cholecystitis	1.00	1.00	1.00	2
Pneumocystosis	1.00	1.00	1.00	1
Dysphagia	1.00	1.00	1.00	1
psychotic disorder	1.00	1.00	1.00	3
hepatitis C	1.00	1.00	1.00	1
CerebroVascular Disease	1.00	1.00	1.00	1
Depression	1.00	1.00	1.00	2
hepatitis	1.00	1.00	1.00	2
hypothyroidism	1.00	1.00	1.00	2
hepatitis B	1.00	1.00	1.00	1
Arthrogryposis	1.00	1.00	1.00	1
manic disorder	1.00	1.00	1.00	2
tonic-clonic seizures	1.00	1.00	1.00	1
migraine	1.00	1.00	1.00	1
anxiety	1.00	1.00	1.00	1
Hepatocellular Carcinoma	1.00	1.00	1.00	1
asthma	1.00	1.00	1.00	2
congestive heart failure	1.00	1.00	1.00	2
hypertensive	1.00	1.00	1.00	1
chronic kidney	1.00	1.00	1.00	1
cirrhosis	1.00	1.00	1.00	1
Blood Coagulation	1.00	1.00	1.00	3
melanoma	1.00	1.00	1.00	1
lymphatic system disease	1.00	1.00	1.00	1
dependence	1.00	1.00	1.00	1
bipolar disorder	1.00	1.00	1.00	1
candidiasis	1.00	1.00	1.00	1
hypoglycemia	1.00	1.00	1.00	1
sinus tachycardia	1.00	1.00	1.00	1
Transient Cerebral Ischemia	1.00	1.00	1.00	3

Figure 6. Values of Precision, recall & F1-score for SVM classifier for each class

6. Conclusion and Future Scope

In this paper, the observations show that the ontology based classification stands at a higher level than the classification without ontology. The general pattern indicates towards a more accurate and precise classification using an ontology. All the parameters that were used (accuracy, precision, recall, and F1 Score) showed an elevation of 1% to 3% when the classification was done with the help of ontology. It can be deduced that using the ontology increased the efficiency of classification. This advantage can be attributed to the fact the number of possible classes for classification reduced while, in turn, reduces the time taken for training purpose. The results also indicate towards a more comparable accuracy level amongst the classifiers when the ontology was used. his study, while proving the importance and benefits of ontology, still has a lot of scope for future improvements. Future work is needed to improve the dataset of diseases used in this project, as there was no official dataset available for the human disease ontology. Most of the work has been done on a limited dataset obtained from converting the available ontology into a dataset. There is also a need to optimize the code used for ontology matching after

the data preprocessing has been done. One could also move on from machine learning towards deep learning and build a neural network for this dataset to further improve the results of the classification in the future.

7. References

1. V. Korde and C. Namrata Mahender, "Text classification and classifiers: a survey," International Journal of Artificial Intelligence & Applications (IJAIA) 3.2 (2012): 85-99.
2. Gelbukh, Natural Language Processing, IEEE Fifth International Conference on Hybrid Intelligent Systems (HIS'05), Rio de Janeiro, Brazil (2006).
3. Agarwal, B. Xie, I. Vovsha, O. Rambow and R. Passonneau, "Sentiment Analysis of Twitter Data (2011). In Proc. WLSM-11.
4. Bhowmick and S.M. Hazarika, E-mail spam filtering: a review of techniques and trends. In: Kalam A, Das S, Sharma K (eds) Advances in electronics, communication and computing. Lecture notes in electrical engineering, 443. Springer, Singapore, 583–590. 2018. https://doi.org/10.1007/978-981-10-4765-7_61
5. S. Akulick and E.S.Mahmoud, "Intent Detection through Text Mining and Analysis. In Proceedings of the Future Technologies Conference (FTC), Vancouver, Canada, 29–30 November 2017; 493–496.
6. K.Das and R.N. Behera, "A Survey on Machine Learning: Concept, Algorithms and Applications," International Journal of Innovative Research in Computer and Communication Engineering 2(2), 2017.
7. Blumauer, PoolParty Semantic Suite, 2018,URL:https://www.poolparty.biz/semantic-ai/
8. https://www.slideshare.net/semwebcompany/semantic-ai
9. T. Berners-Lee, James Hendler and Ora Lassila, Scientific American: Feature Article: The Semantic Web: May 2001
10. K.A. Vidhya, G. Aghila, "A Survey of Naïve Bayes Machine Learning approach in Text Document Classification", (IJCSIS) International Journal of Computer Science and Information Security, 7, 2010.
11. T. JOACHIMS, Text categorization with support vector machines: learning with many relevant features. In Proceedings of ECML-98, 10th European Conference on Machine Learning (Chemnitz, Germany, 1998), 137-142.
12. D. E. Johnson, F. J. Oles, T. Zhang, T. Goetz, "A decision-tree-based symbolic rule induction system for text Categorization", by IBM systems journal, 41(3) 2002.
13. A.A. Salatino, T. Thanapalasingam, A. Mannocci, F. Osborne and E. Motta, "Classifying Research Papers with the Computer Science Ontology," Knowledge Media Institute, The Open University, MK7 6AA, Milton Keynes, UK, 2018.
14. A.A. Salatino, F. Osborne and E. Motta, "The Computer Science Ontology: A Large-Scale Taxonomy of Research Areas," 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part II
15. A. Salatino, F. Osborne, E. Motta. The CSO classifier: Ontology-driven detection of research topics in scholarly articles. In: A. Doucet et al. (eds.) TPDL 2019: 23rd International Conference on Theory and Practice of Digital Libraries. Cham, Switzerland: Springer, 2019, pp. 296–311. doi: 10.1007/978- 3-030-30760-8_26.
16. J. Ma, W. Xu, Y. Sun, E. Turban, S. Wang, O. Liu, "An Ontology-Based Text-Mining Method to Cluster Proposals for Research Project Selection," IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans, 42(3), 2012.
17. P. Kaur, R. Sapra, "Ontology Based Classification and Clustering of Research Proposals and External Research Reviewers", International Journal of Computers & Technology, 5(1) 2013, ISSN 2277-3061
18. C.M. Wijewickrema, R. Gamage, An Ontology Based Fully Automatic Document Classification System Using an Existing Semi-Automatic System, National Institute of Library and Information Sciences, University of Colombo, Colombo, Sri Lanka, 2013.
19. Sudha Ramkumar, B. Poorna, B. Saleena, Ontology based text document clustering for sports, Journal of Engineering and Applied Sciences, 2018.
20. Nayat Sanchez-Pi, Luis Marti and A.C.B. Garcia, "Improving ontology-based text classification: An occupational health and security application," Article in Journal of Applied Logic · September 2015
21. S.L. Decker, B. Aleman-meza, D. Cameron and I.B. Arpinar, Detection of Bursty and Emerging Trends towards Identification of Researchers, the Early Stage of Trends (2007).
22. M. Herrera, D.C. Roberts and N. Gulbahce, Mapping the evolution of scientific fields. PLoS ONE. 5(5), 2010.
23. R.L. Ohniwa, A. Hibino and K. Takeyasu, Trends in

research foci in life science fields over the last 30 years monitored by emerging topics, Scientometrics. 85(1), 2010.
24. F. Mai, L. Galke, and A. Scherp, Using Deep Learning for Title Based Semantic Subject Indexing to Reach Competitive Performance to Full-Text, JCDL '18 Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries (Fort Worth, Texas, USA, Jun. 2018)
25. W. D. Cook, B. Golany, M. Kress, M. Penn, and T. Raviv, "Optimal allocation of proposals to reviewers to facilitate effective ranking," Manage. Sci., 51(4), 655–661, 2005.
26. Arya and B. Mittendorf, "Project assignment when budget padding taints resource allocation," Manage. Sci., vol. 52, no. 9, pp. 1345–1358, Sep. 2006.
27. Choi and Y. Park, "R&D proposal screening system based on text mining approach," Int. J. Technol. Intell. Plan, 2(1), 61–72, 2006.
28. K. Girotra, C. Terwiesch, and K. T. Ulrich, "Valuing R&D projects in a portfolio: Evidence from the pharmaceutical industry," Manage. Sci., 53(9) 1452–1466, 2007.
29. Y. H. Sun, J. Ma, Z. P. Fan, and J. Wang, "A group decision support approach to evaluate experts for R&D project selection," IEEE Transactions of Engineering management, 55(1), 158–170, 2008.
30. Y. H. Sun, J. Ma, Z. P. Fan, and J. Wang, "A hybrid knowledge and model approach for reviewer assignment," Expert System Applications, 34(2), 817–824, Feb. 2008.
31. M. Allahyari, K. J. Kochut and M. Janik, Ontology-based text classification into dynamically defined topics, Semantic Computing (ICSC), 273-278, 2014.
32. R. Prabowo, M. Jackson, P. Burden and H. Knoell, Ontology-Based Automatic Classification for the Web
40. G. Guo, H. Wang, D. Bell, Y. Bi and K. Greer, KNN Model-Based Approach in Classification, Proc. ODBASE pp- 986 – 996, 2003
41. G. Biau, "Analysis of a Random Forests Model", Journal of Machine Learning Research 13 (2012) 1063-1095
42. Y. Qin, X. Wang, Study on Multi-label Text Classification Based on SVM, Sixth International Conference on Fuzzy Systems and Knowledge Discovery 2009
43. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingClassifier.html
44. https://bioportal.bioontology.org/ontologies/DOID
45. http://people.dbmi.columbia.edu/~friedma/Projects/DiseaseSymptomKB/index.html

Pages Design Implementation and Evaluation", Proc. Of the 3rd International Conference on Web Information Systems Engineering, 2002.
33. F.Y. Osisanwo, J.E.T. Akinsola, O. Awodele, J.O. Hinmikaiye, O. Olakanmi and J. Akinjobi Supervised Machine Learning Algorithms: Classification and Comparison, International Journal of Computer Trends and Technology (IJCTT), 48(3), 2017.
34. M. Khanum, T. Mahboob, W. Imtiaz, H.A. Ghafoor and R. Sehar, A Survey on Unsupervised Machine Learning Algorithms for Automation, Classification and Maintenance, International Journal of Computer Applications (0975 – 8887) 119(13), 2015.
35. Q. Guo, J. Wentian, S. Zhong and E. Zhou, "The Analysis of the Ontology-based K-Means Clustering Algorithm", Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013), [online] Available: https://www.atlantis-press.com/proceedings/iccsee-13/4617.
36. P. Vateekul and M. Kubat, Fast Induction of Multiple Decision Trees in Text Categorization From Large Scale,Imbalanced, and Multi-label Data, IEEE International Conference on Data MiningWorkshops 2009.
37. S. Dargan, M. Kumar, M. Ayyagari and G. Kumar, A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning, Springer, June 2019.
38. https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html
39. S. Xu, Y. Li and Z. Wang, Bayesian multinomial naïve bayes classifier to text classification. In: Advanced multimedia and ubiquitous engineering. Springer, 347–352, 2017.

46. Y. Freund and R.E. Schapire, A Short Introduction to Boosting" Journal of Japanese Society for Artificial Intelligence, 14(5), 771-780, 1999.
47. V.N. Garla, C. Brandt, Ontology-Guided Feature Engineering for Clinical Text Classification, Journal of Biomedical Informatics, 45(5): 992–998. doi: 10.1016/j.jbi.2012.04.010
48. https://github.com/nltk/nltk
49. D. Fensel. Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce. Springer-Verlag, 2001.
50. https://www.forbes.com/sites/forbestechcouncil/2019/12/30/explainable-ai-the-rising-role-of-knowledge scientists/#62bc6193603f