TEXT MINING OF SCIENTIFIC PUBLICATIONS

A PROJECT REPORT

Submitted by

Srinivas T R 2017115601

Rithvik A V S 2017115579

submitted to the Faculty of

INFORMATION AND COMMUNICATION ENGINEERING

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

COLLEGE OF ENGINEERING, GUINDY

ANNA UNIVERSITY

CHENNAI 600 025

ANNA UNIVERSITY

CHENNAI - 600 025

BONA FIDE CERTIFICATE

Certified that this project report Text mining of Scientific Publications is the bona fide work of Srinivas T R, Rithvik A.V.S who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE:CHENNAI
DATE:23/11/2020

Dr.Saswathi Mukherjee
PROFESSOR
PROJECT GUIDE
DEPARTMENT OF IST, CEG
ANNA UNIVERSITY
CHENNAI 600025

COUNTERSIGNED

Dr. SASWATI MUKHERJEE
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT

The amount of research work taking place in all streams of Science, Engineering, Medicines, etc., is growing rapidly and hence the research articles that are being published are increasing day by day. In this dynamic environment, identifying and maintaining such a large collection of articles in one place and classifying them manually is becoming very exhaustive. Finding scientific papers and journals relevant to a particular area of research is a concern for many people including students, professors, and researchers. Classification of papers into the relevant categories facilitates the search process. This task can be done manually by, for example, asking authors to assign one or more categories during publishing time. However, categorizing a large collection of resources manually is a time consuming process. In automatic methods, a naive strategy is to do a keyword-based search for the subject term in paper's title, keywords, and even its fulltext. Nonetheless, this approach fails for resources employing semantically equivalent terms but not exactly the same subject words. Besides, processing the whole text of a paper takes a long time. This project proposes an approach for classifying such huge volume of articles by using machine learning classification methods. Since one research paper might belong to different categories simultaneously, the need for multi label classification arises. Various approaches to feature extraction, multi label feature selection and multi label classification are studied and implemented.

TAMIL ABSTRACT

அறிவியல், பொறியியல், மருந்துகள் போன்ற அனைத்து நீரோடைகளிலும் நடைபெற்று வரும் ஆராய்ச்சிப் பணிகளின் அளவு வேகமாக வளர்ந்து வருகிறது, எனவே வெளியிடப்படும் ஆய்வுக் கட்டுரைகள் நாளுக்கு நாள் அதிகரித்து வருகின்றன. இந்த மாறும் சூழலில், இவ்வளவு பெரிய கட்டுரைகளை ஒரே இடத்தில் கண்டறிந்து பராமரிப்பது மற்றும் அவற்றை கைமுறையாக வகைப்படுத்துவது மிகவும் முழுமையானதாகி வருகிறது. ஒரு குறிப்பிட்ட பகுதிக்கு பொருத்தமான விஞ்ஞான ஆவணங்கள் மற்றும் பத்திரிகைகளைக் கண்டுபிடிப்பது மாணவர்கள், பேராசிரியர்கள் மற்றும் ஆராய்ச்சியாளர்கள் உட்பட பலருக்கு கவலை அளிக்கிறது. ஆவணங்களை தொடர்புடைய வகைகளாக வகைப்படுத்துவது தேடல் செயல்முறைக்கு உதவுகிறது. எடுத்துக்காட்டாக, வெளியீட்டு நேரத்தில் ஒன்று அல்லது அதற்கு மேற்பட்ட வகைகளை ஒதுக்குமாறு ஆசிரியர்களைக் கேட்டு இந்த பணியை கைமுறையாக செய்ய முடியும். இருப்பினும், ஒரு பெரிய வளங்களை கைமுறையாக வகைப்படுத்துவது நேரத்தை எடுத்துக்கொள்ளும் செயல்முறையாகும். தானியங்கி முறைகளில், காகிதத்தின் தலைப்பு, முக்கிய சொற்கள் மற்றும் அதன் முழு உரையில் கூட பொருள் காலத்திற்கான முக்கிய சொற்களைத் தேடுவது ஒரு அப்பாவி உத்தி. ஆயினும்கூட, இந்த அணுகுமுறை சொற்பொருளுக்கு சமமான சொற்களைப் பயன்படுத்தும் வளங்களுக்கு தோல்வியடைகிறது, ஆனால் அதே பொருள் சொற்கள் அல்ல. தவிர, ஒரு காகிதத்தின் முழு உரையையும் செயலாக்க நீண்ட நேரம் எடுக்கும். இயந்திர கற்றல் வகைப்பாடு முறைகளைப் பயன்படுத்தி இத்தகைய பெரிய அளவிலான கட்டுரைகளை வகைப்படுத்துவதற்கான அணுகுமுறையை இந்த திட்டம் முன்மொழிகிறது. ஒரு ஆய்வுக் கட்டுரை ஒரே நேரத்தில் வெவ்வேறு வகைகளைச் சேர்ந்ததாக இருப்பதால், பல லேபிள் வகைப்பாட்டின் தேவை எழுகிறது. அம்சம் பிரித்தெடுத்தல், மல்டி லேபிள் அம்சத் தேர்வு மற்றும் பல லேபிள் வகைப்பாடு ஆகியவற்றிற்கான பல்வேறு அணுகுமுறைகள் ஆய்வு செய்யப்பட்டு செயல்படுத்தப்படுகின்றன.

ACKNOWLEDGEMENT

First and foremost, we would like to express our deep sense of gratitude to our guide **Dr.Saswati Mukherjee**, Professor Department of Information Science and Technology, Anna University for their excellent guidance,counsel,continuous support and patience. They helped us with this topic and guided us in the development of this project. They gave us the moral support to finish out creative and innovative project in a successful manner.

We would also like to express our gratitude to **Dr.Saswati Mukherjee**, Head of the Department, Department of Information Science and Technology,Anna University,for supporting us with the technical resources required for our project. We express our heartiest thanks to all the other teaching and non teaching staffs who have helped us in the successful completion of the project. We would also like to thank our parents and friends for their indirect contribution in the successful completion of the project.

SRINIVAS T R

RITHVIK A V S

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This chapter gives an overview of the text mining, multi label classification and the research that has been done over the last few years. Furthermore this briefly explains the motive, challenges faced in developing this method.

1.1 Text Mining

Research in text mining has become one of the most widespread fields in analysing the natural language documents. Currently, almost every existing information from different institutions (e.g. government, business, industry, and others) is preserved in electronic documents which is in the form of semi-structured data. Therefore, the need for text mining arises as text mining is different from data mining. Data mining is focused on discovering interesting patterns from large databases rather than textual information. Text mining intends to detect the information that was not recognized before through extracting it automatically from various text-based sources. Structured data can be handled through data mining tools while unstructured or semi-structured datasets like full-text documents, emails, and HTML files can be handled efficiently through text mining. The common structure of text mining involves two consecutive stages: text refining and knowledge distillation. In text refining, free-form text documents are converted into an intermediate form, whereas in knowledge distillation, patterns or knowledge are derived from intermediate form.

1.2 Multi Label Classification

In Machine Learning, and particularly in supervised learning, classification is one the most important learning techniques. In the traditional task of singlelabel classification each example is associated with a single class label. A classifier learns to associate each new test example with one of these known class labels. When each example may be associated with multiple labels, this is known as multi-label classification. Multilabel classification is a challenging research problem that emerges in several modern applications such as music categorization, protein function classification and semantic classification of images. In the past, multilabel classification has mainly engaged the attention of researchers working on text categorization, as each member of a document collection usually belongs to more than one semantic category. Multilabel classification methods can be categorized into two different groups

- Problem transformation methods
- Algorithm adaptation methods.

The first group of methods are algorithm independent. They transform the multilabel classification task into one or more single-label classification, regression or label ranking tasks. The second group of methods extend specific learning algorithms in order to handle multilabel data directly.

1.3 Motivation

There has been a limitation in categorizing research articles automatically. Most of the categorization is based on just the keywords present in the research papers. These are very inefficient. And also, classifying archived research

papers into categories manually is very inefficient and time consuming task.

1.4 Problem Statement

A research paper may belong to more than one category at the same time. In existing literature, many text mining and machine learning classification methods have been applied for classifying research documents. But there is a limitation in addressing multi label classification of scientific documents.

1.5 Objective

The project aims to find an appropriate and efficient method for multi label text classification of scientific documents.

1.6 Organisation of the report

This report explains the proposed methodologies,tools and the resultant outcome of the project.

CHAPTER 2

LITERATURE SURVEY/RELATED WORK

Following papers have been referred to gain insights into the existing methods for clustering and multi-label classification.

2.1 Classification of Scientific Text

2.1.1 Research paper classifcation systems based on TF-IDF and LDA Schemes

The proposed system extracts representative keywords from the abstracts of each paper and topics by Latent Dirichlet allocation (LDA) scheme [1]. Then, the K-means clustering algorithm is applied to classify the whole papers into research papers with similar subjects, based on the Term frequency-inverse document frequency (TF-IDF) values of each paper.

2.1.2 SCIBERT

SCIBERT, a pretrained language model based on BERT addresses the lack of high-quality, large-scale labeled scientific data [2]. SCIBERT leverages unsupervised pretraining on a large multi-domain corpus of scientific publications to improve performance on downstream scientific NLP tasks.

2.1.3 Summarization and clustering

The method proposed is a summarization-based hybrid algorithm which comprises a preprocessing phase. In the summarization phase unimportant words which are not frequently used in the document are removed. This process reduces the amount of data for the clustering purpose. In this proposed method after the preprocessing phase, Term Frequency/Inverse Document Frequency (TFIDF) is calculated for all words in the document and BM25 in calculated for words in sentences and summed over the document to score each word in document level [3]. In next phase, Text summarization is performed based on BM25 scores. After that document clustering is done according to the scores of calculated TFIDF. The hybrid progress of the proposed scheme, from preprocessing phase to cluster labeling, gains a rapid and efficient clustering method which is evaluated by 400 English texts extracted from scientific articles of 11 different topics.
The result of this clustering has shown that the new method of using text summarization for eliminating not useful words has been effective in order to perform effective clustering. This method has gained incredibly low running time compare to all comparative methods

2.2 Multi-Lablel Classification

2.2.1 Random K label subsets

This paper proposes an ensemble method for multilabel classification. The RAndom k-labELsets (RAKEL) algorithm constructs each member of the ensemble by considering a small random subset of labels and learning a single-label classifier for the prediction of each element in the powerset of this subset.[4]
In this way, the proposed algorithm aims to take into account label correlations using single-label classifiers that are applied on subtasks with manageable number of labels and adequate number of examples per label. Experimental results on

common multilabel domains involving protein, document and scene classification show that better performance can be achieved compared to popular multilabel classification approaches.

2.2.2 Feature Selection on Multi label Text classification

Multi-label text classification deals with problems in which each document is associated with a subset of categories. These documents often consist of a large number of words, which can hinder the performance of learning algorithms. Feature selection is a popular task to find representative words and remove unimportant ones, which could speed up learning and even improve learning performance[5]. This work evaluates eight feature selection algorithms in text benchmark datasets.The best algorithms are subsequently compared with random feature selection and classifiers built using all features.
Results agree with literature by finding that well-known approaches, such as maximum chi-squared scoring across all labels, are good choices to reduce text dimensionality while reaching competitive multi-label classification performanc

2.2.3 Classifier Chains for Multi-label Classification

The widely known binary relevance method for multi-label classification, considers each label as an independent binary problem. Their work showed that binary relevance-based methods have much to offer, especially in terms of scalability to large datasets [6]. A novel chaining method was developed that can model label correlations while maintaining acceptable computational complexity. Empirical evaluation over a broad range of multi-label datasets with a variety of evaluation metrics demonstrated the competitiveness of the chaining method against related and state-of-the-art methods, both in terms of predictive

performance and time complexity.
By passing label correlation information along a chain of classifiers, their method counteracted the disadvantages of the binary method while maintaining acceptable computational complexity. An ensemble of classifier chains can be used to further augment predictive performance. Using a variety of multi-label datasets and evaluation measures, they carried out empirical evaluations against a range of algorithms. The classifier chains method proved superior to related methods, and in an ensemble scenario was able to improve on state-of-the-art methods, particularly on large datasets. Despite other methods using more complex processes to model label correlations, ensembles of classifier chains can achieve better predictive performance and are efficient enough to scale up to very large problems.

2.2.4 Ml-KNN

A lazy learning algorithm named Ml-knn, which is the multi- label version of kNN, is proposed[7]. Based on statistical information derived from the label sets of an unseen instance's neighboring instances, Ml-knn utilizes maximum a posteriori principle to determine the label set for the unseen instance. Experiments on three real-world multi-label learning problems, i.e. Yeast gene functional analysis, natural scene classification and automatic web page categorization, show that Ml-knn outperforms some well-established multi-label learning algorithms. In this paper, the distance between instances is simply measured by Euclidean metric.

CHAPTER 3

DESIGN

For multilabel text classification the following modules are used.

- Data Wrangling
- Text preprocessing
- Feature Extraction
- Feature Selection
- Classification
- Hyperparameter Tuning and Regularization
- Evaluation and comparison of performance and accuracy.

3.1 Overall Flow Diagram

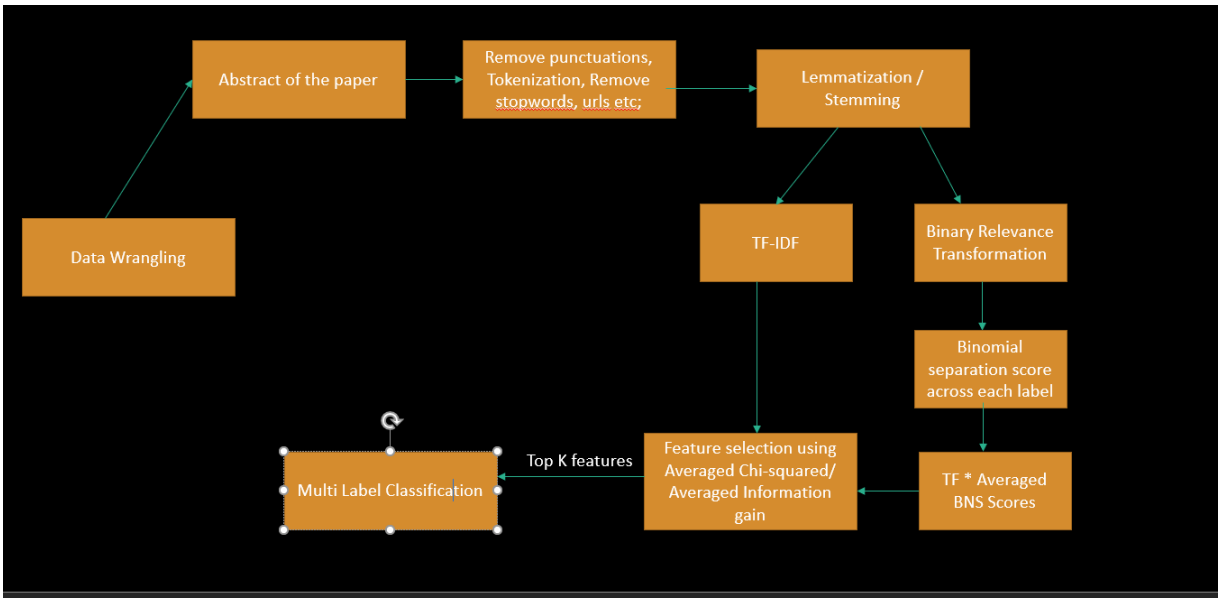


Figure 3.1: Overall flow diagram

Fig 3.1 shows the overall flow diagram for multi-label classification of scientific publications.

3.2 Data Wrangling

In the data wrangling phase, the data is acquired, required fields are chosen and converted into a comma separated value file format. The dataset is also sampled.

3.3 Text Preprocessing

Text preprocessing consists of the following stages:

- Removing punctuations, whitespaces, URLs,numbers etc;
- Tokenization
- Stopwords removal

A stop word is a commonly used word such as “the”, “a”, “an”, “in” etc;. These are removed from the sentence after tokenization. Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item. Lemmatization is similar to stemming but it brings context to the words. So it links words with similar meaning to one word.

3.4 Feature Extraction

Abstract is chosen as the feature for multi label classification. Two different feature extraction methods are used to convert the text into numeric format for classification.

3.4.1 Term Frequency - Inverse document frequency

Tf-idf is a weighting scheme that assigns each term in a document a weight based on its term frequency (tf) and inverse document frequency (idf). The terms with higher weight scores are considered to be more important.

Typically, the tf-idf weight is composed by two terms-

- Normalized Term Frequency (tf)
- Inverse Document Frequency (idf)

$$tf(t,d) = N(t,d) \tag{3.1}$$

wherein tf(t, d) = term frequency for a term t in document d. N(t, d) = number of times a term t occurs in document d.

$$tf(t,d) = N(t,d)/||D|| \tag{3.2}$$

wherein,D= Total number of terms in the document

$$df(t) = N(t) \tag{3.3}$$

where df(t) = Document frequency of a term t and N(t) = Number of documents containing the term /

$$idf(t) = N/df(t) = N/N(t) \tag{3.4}$$

$$idf(t) = log(N/df(t)) \tag{3.5}$$

$$tfidf(t,d) = tf(t,d) * idf(t,d) \tag{3.6}$$

3.5 Term Frequency - Binomial Separation

Bi-Normal Separation is denoted by: $| F^{-1}(tpr) - F^{-1}(fpr) |$ where,

$$true\ positive\ rate(tpr) = P(word \mid positiveclass) = tp/pos \tag{3.7}$$

$$false\ positive\ rate(fpr) = P(word \mid negativeclass) = fp/neg \tag{3.8}$$

F^{-1} is the inverse Normal cumulative distribution function, as commonly available from statistical tables. Since BNS can be applied to only binary problems, the multi label problem is transformed into a binary problem where q refers to the total number of distinct labels in the dataset. This is known as binary relevance transformation. BNS Scores are calculated for each term in the corpus and the average across q labels is taken. The BNS scores are then multiplied with term frequency vectors.

$$tfbns(t,d) = tf(t,d) * bns(t) \tag{3.9}$$

3.6 Feature Selection

TF-IDF and TF-BNS vectors consist of a large number of features which is the total number of unique words in the document. Out of this, the best k features can be applied by using feature selection algorithms. Chi squared feature Selection and decision tree based feature selection are used.

Since these algorithms can be directly applied only on binary problems binary relevance transformation is used to convert the problem into q binary problems and the average of feature scores across these q problems are taken.

3.6.1 Chi-Squared feature Selection:

A chi-square test is used in statistics to test the independence of two events. Given the data of two variables, we can get observed count O and expected count E. Chi-Square measures how expected count E and observed count O deviates each other. So high Chi-Square value indicates that the hypothesis of independence is incorrect. Higher the Chi-Square value the feature is more dependent on the response and it can be selected for model training. It is denoted by

$$\tilde{\chi}^2 = \frac{1}{d}\sum_{k=1}^{n}\frac{(O_k - E_k)^2}{E_k} \tag{3.10}$$

where:

- O = Observed Value
- E = Expected Value

3.6.2 Decision tree based feature selection

Tree based models calculates feature importance for they need to keep the best performing features as close to the root of the tree. Constructing a decision tree involves calculating the best predictive feature. The feature importance in tree based models are calculated based on Information Gain or splits.

3.7 Multi Label text Classification

The following methods are used to perform multi label text classification.

- Multi Label K-Nearest Neighbours
- Classifier Chains
- Random K Label subsets

The following classifiers are used along with problem transformation approaches:

3.7.1 LinearSVC

LinearSVC is a linear support-vector machine that uses liblinear solver to optimize the problem.

3.7.2 SGDClassifier

SGDClassifier is a support vector machine which has linear kernel by default and uses Stochastic Gradient Descent that is widely used in neural networks to optimize the problem.

3.7.3 Gradient Boosting Classifier

Boosting is an ensemble technique in which the predictors are not made independently, but sequentially. This technique employs the logic in which the subsequent predictors learn from the mistakes of the previous predictors.

3.7.4 LightGBM

Light GBM is a gradient boosting framework released by microsoft that uses tree based learning algorithm. Light GBM grows tree vertically while other algorithm grows trees horizontally meaning that Light GBM grows tree leaf-wise while other algorithm grows level-wise.

CHAPTER 4

IMPLEMENTATION

4.1 Tools Used

The following tools, libraries and environments are used in this project

4.1.1 Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license [8].

4.1.2 Dask

Dask is an open source library for parallel computing written in Python [9].

4.1.3 Scikit Learn

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines. [10]

4.1.4 Scikit Multilearn

Scikit-multilearn is a BSD-licensed library for multi-label classification that is built on top of the well-known scikit-learn ecosystem.[11]

4.1.5 NLTK

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing for English written in the Python programming language. [12]

4.1.6 Swifter

Swifter is a package which efficiently applies any function to a pandas dataframe or series in the fastest available manner.

4.1.7 Google Colab Notebooks

Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education

4.2 Dataset Used

Arxiv is a repository of 1.7 million articles, with relevant features such as article titles, authors, categories, abstracts, full text PDFs, and more [13]. This dataset is a mirror of the original ArXiv data. Because the full dataset

is rather large (1.1TB and growing), this dataset provides only a metadata file in the json format. This file contains an entry for each paper, containing:

- id: ArXiv ID (can be used to access the paper, see below)
- submitter: Who submitted the paper
- authors: Authors of the paper
- title: Title of the paper
- comments: Additional info, such as number of pages and figures
- journal-ref: Information about the journal the paper was published in
- doi: [https://www.doi.org](Digital Object Identifier)
- abstract: The abstract of the paper
- categories: Categories / tags in the ArXiv system
- versions: A version history

The dataset has 1704576 instances in total. 70% of the data is used for training and 30% for testing.

4.3 Data Wrangling

4.3.1 Format Conversion

Data Wrangling is done using dask. Dask is a flexible library for parallel computing in Python. The dataset is acquired in Javascript Object Notation format. Since loading entire JSON object takes a huge amount of RAM dask is used. Dask splits the dataset into various chunks and only loads the necessary chunk into memory. The following fields are chosen and the resultant data is converted into comma separated value file format. This has 1704576 rows.

4.3.2 Label Cleaning

In the dataset being used, a row may belong to more than one category. Each category also contains a subcategory which is separated by a dot. These subcategories are removed for making the classification easier. After this phase, the dataset has 31 distinct labels.

4.3.3 Sampling

Proportionate stratified random sampling is used to sample the dataset. 10% of samples from each distinct label is chosen to maintain class balance. From this label, a subset that contains only the combination of top 11 labels is chosen so as to avoid space and time complexity issues during classification. At the end, 131549 rows are left after sampling.

	id	title	categories	abstract
0	0704.0001	Calculation of prompt diphoton production cros...	hep	A fully differential calculation in perturba...
1	0704.0002	Sparsity-certifying Graph Decompositions	math cs	We describe a new algorithm, the (k,ℓ)-...
2	0704.0003	The evolution of the Earth-Moon system based o...	physics	The evolution of Earth-Moon system is descri...
3	0704.0004	A determinant of Stirling cycle numbers counts...	math	We show that a determinant of Stirling cycle...
4	0704.0005	From dyadic Λ_{α} to $\Lambda_{\a...	math	In this paper we show how to compute the $\L...
...	...	...	...	...
1788988	quant-ph/9912118	A Fast and Compact Quantum Random Number Gener...	quant	We present the realization of a physical qua...
1788989	quant-ph/9912119	Nuclear Teleportation	quant	Until recently, only science-fiction authors...
1788991	quant-ph/9912121	Simple pulse sequence for quantum logic operat...	quant	This paper has been withdrawn by the authors...
1788992	quant-ph/9912122	Optimal signal ensembles	quant	Classical messages can be sent via a noisy q...
1788993	quant-ph/9912123	Multilevel Quantum Particle as a Few Virtual Q...	quant	A conception of virtual quantum information ...

1704576 rows × 4 columns

Figure 4.1: Dataset
Dataset in csv format

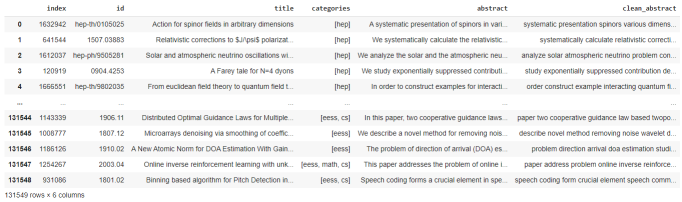
```
Index(['id', 'title', 'categories', 'abstract'], dtype='object') {'hep': 297543, 'math': 458395, 'cs': 301330, 'physics': 162153, 'cond': 282717, 'gr': 84116,
'astro': 278635, 'nucl': 60040, 'quant': 105561, 'stat': 76276, 'eess': 26807}
```

Figure 4.2: Top 11 categories
Subset is chosen from the dataset such that it consists of combination of these 11 categories.

4.4 Data Preprocessing

- Removal of digits is done by using regular expression replacement.
- Punctuations are replaced by constructing a lookup table.
- NLTK tokenizer is used to tokenize the data
- NLTK Wordnet Lemmatizer is used to lemmatize the words after tokenization.
- In the tokenized list, all other words except the stopwords are combined to form a string.

4.5 Feature Extraction



	index	id	title	categories	abstract	clean_abstract
0	1632942	hep-th/0105025	Action for spinor fields in arbitrary dimensions	[hep]	A systematic presentation of spinors in vari...	systematic presentation spinors various dimens...
1	641544	1507.03883	Relativistic corrections to J/ψ polarizat...	[hep]	We systematically calculate the relativistic...	systematically calculate relativistic correcti...
2	1612037	hep-ph/9505281	Solar and atmospheric neutrino oscillations wi...	[hep]	We analyze the solar and the atmospheric neu...	analyze solar atmospheric neutrino problem con...
3	120919	0904.4253	A Farey tale for N=4 dyons	[hep]	We study exponentially suppressed contributi...	study exponentially suppressed contribution de...
4	1666551	hep-th/9802035	From euclidean field theory to quantum field t...	[hep]	In order to construct examples for interacti...	order construct example interacting quantum fi...
...	...	...	...	...	...	...
131544	1143339	1906.11	Distributed Optimal Guidance Laws for Multiple...	[eess, cs]	In this paper, two cooperative guidance laws...	paper two cooperative guidance law based twopo...
131545	1008777	1807.12	Microarrays denoising via smoothing of coeffic...	[eess]	We describe a novel method for removing nois...	describe novel method removing noise wavelet d...
131546	1186126	1910.02	A New Atomic Norm for DOA Estimation With Gain...	[eess]	The problem of direction of arrival (DOA) es...	problem direction arrival doa estimation studi...
131547	1254267	2003.04	Online inverse reinforcement learning with unk...	[eess, math, cs]	This paper addresses the problem of online i...	paper address problem online inverse reinforce...
131548	931086	1801.02	Binning based algorithm for Pitch Detection in...	[eess, cs]	Speech coding forms a crucial element in spe...	speech coding form crucial element speech comm...

131549 rows × 6 columns

Figure 4.3: Sample Dataset
Dataset after sampling

abstract	clean_abstract
A systematic presentation of spinors in vari...	systematic presentation spinors various dimens...
We systematically calculate the relativistic...	systematically calculate relativistic correcti...
/e analyze the solar and the atmospheric neu...	analyze solar atmospheric neutrino problem con...
We study exponentially suppressed contributi...	study exponentially suppressed contribution de...
In order to construct examples for interacti...	order construct example interacting quantum fi...
...	...
In this paper, two cooperative guidance laws...	paper two cooperative guidance law based twopo...
/e describe a novel method for removing nois...	describe novel method removing noise wavelet d...
The problem of direction of arrival (DOA) es...	problem direction arrival doa estimation studi...
This paper addresses the problem of online i...	paper address problem online inverse reinforce...
Speech coding forms a crucial element in spe...	speech coding form crucial element speech comm...

Figure 4.4: Clean Abstract
Abstract after preprocessing.

4.5.1 Feature Extraction from Label

Scikit Learn's MultiLabelBinarizer module is used to convert an array of labels to a vector of dimension 131549 * 11 where each column represents a single label and if a row belongs to a particular label it is denoted as "1" else "0".

4.5.2 Feature Extraction from abstract

TF-IDF and TF-BNS are used for feature extraction. Scikit-learn's

```
[['systematic presentation spinors various dimension given']
 ['analyze solar atmospheric neutrino problem context three flavour neutrino osci
 ['order construct example interacting quantum field theory model method euclidea
 ...
 ['problem direction arrival doa estimation studied decade essential technology e
 ['paper address problem online inverse reinforcement learning nonlinear system m
 ['speech coding form crucial element speech communication important area concern
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 1 ... 0 0 0]
 [0 0 1 ... 0 0 0]]
```

Figure 4.5: Multi label binarization
List of labels is converted into a matrix of 131459 * 11 dimensions

TfIDfVectorizer is used to convert the abstract into a TF-IDF vector.

TF-BNS is implemented from scratch. Swifter is added as a wrapper function over pandas to enable multi core parallelisation during calculations. Once BNS scores are computed,it is converted into a sparse diagonal matrix

```
{'aa': 0.39236835474680926,
 'aaa': 1.183239185883297,
 'aaaa': 1.2230371489338847,
 'aaaaa': 1.0691748886853873,
 'aaaaaa': 1.2230371489338847,
 'aaab': 1.3394596633830445,
 'aaac': 1.2467366176044397,
 'aaaldotsak': 1.0691748886853873,
 'aaan': 1.38219287949917,
 'aaanaan': 1.0691748886853873,
 'aaastacked': 1.091094888925499,
 'aab': 0.691470792785723,
 'aabb': 1.0924028104904822,
 'aabbaabb': 1.083647788672346,
 'aabbab': 1.2467366176044397,
 'aabh': 1.4447724444860848,
 'aabin': 1.0691748886853873,
 'aablg': 1.338106072298362,
 'aabridged': 1.0924028104904822,
 'aabstacked': 1.091094888925499,
 'aac': 1.1578749959132102,
 'aacalphafrac': 1.2230371489338847,
 'aacdotsan': 1.2230371489338847,
 'aachen': 1.0690374100645703,
 'aacollisions': 1.1832179666674359,
 'aacr': 1.091094888925499,
 'aad': 0.6070808963749493,
 'aada': 1.1785196906849493,
 'aadbased': 1.1496626945274082,
 'aadebug': 1.3931598866964992,
 'aadequate': 1.3154603135716696,
```

Figure 4.6: Binomial Seperation Scores
Bionomial seperation scores of a few features sorted in alphabetical order

and a dot product between Term Frequency vector and Diagonal BNS vector is taken to form the TF-BNS matrix. When a simple for loop was used to construct the TF-BNS matrix the running time was approximately 30 minutes. After

vectorization, the running time significantly dropped to approximately 1 minute.

```
(0, 179664)   0.17037500970141128
(0, 165035)   0.25707265646163735
(0, 157788)   1.0330758760342968
(0, 131932)   0.27694360339682544
(0, 62649)    0.280701663097668
(0, 38147)    0.3751392063177144
(1, 179138)   0.9935688229433611
(1, 178492)   0.44355868138129073
(1, 169799)   0.2139739361775
(1, 164216)   0.23030794607902585
(1, 162201)   0.19684834857636482
(1, 157201)   0.4638151707618919
(1, 156546)   2.2273742145820794
(1, 156155)   0.1641630144887273
(1, 155940)   1.5261132593799338
(1, 154886)   0.22003016413969856
(1, 151961)   1.0331830585007273
(1, 151622)   0.25793793814028726
(1, 147392)   0.49959655775343265
(1, 146905)   0.2024234701574992
(1, 142609)   0.28524409303747805
(1, 141030)   1.156182483547813
(1, 140984)   0.14976446000765273
(1, 140197)   1.0555689730403481
(1, 132578)   1.5309240909738167
:     :
(92043, 46825)        0.1430905789830578
(92043, 45350)        0.27345548582947016
(92043, 39495)        0.2918647705148874
(92043, 37573)        0.9566685730424674
(92043, 36806)        0.613273774814707
(92043, 36794)        0.27578959169837575
(92043, 36585)        0.24132192171558167
(92043, 31545)        0.12155154003987555
(92043, 28100)        0.31056287362015494
(92043, 27826)        0.2171918981437486
(92043, 27246)        0.17768614213609485
(92043, 27091)        0.6441732332746323
(92043, 26920)        0.16821076367665916
(92043, 26035)        0.9919013990044789
(92043, 24607)        0.9336744747656058
(92043, 24587)        0.3525046002102497
(92043, 13923)        0.6943085333721889
```

Figure 4.7: TF-BNS
Term Frequency - Binomial Seperation vector. Since this is represented as a sparse matrix only the fields which are not zero are stored in the memory.

4.6 Feature Selection

4.6.1 Chi squared Feature Selectiom

Across each of the 11 labels, chi square score is calculated for each feature from the feature matrix. To calculate Chi2 square SelectKBest module from Scikit Learn is used. Top k features are chosen. The performance of classification is evaluated with k = 2000 and k = 4000.

4.6.2 Feature Selection using LightGBM

LightGBM decision trees are used to select top k features based on split values. Split contains the number of times a feature is used to split the data across all trees in the ensemble of decision trees. The features with top k values are chosen.The performance of classification is evaluated with k = 2000 and k = 4000.

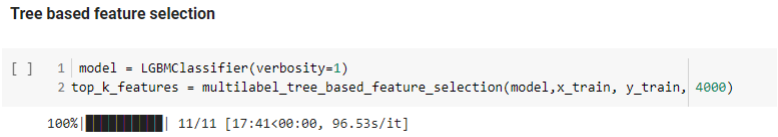


Figure 4.8: Tree based feature selection
Top k features are chosen based on split value after constructing a LightGBM tree.

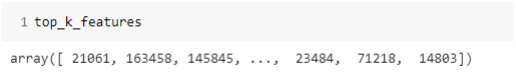


Figure 4.9: Indices of top 2000 features
The list contains column number of top 2000 features in the TF-BNS matrix

4.7 Metrics for evaluating multi label classification

4.7.1 Hamming Loss

Hamming loss is the fraction of labels that are incorrectly predicted for a single sample. To find the total hamming loss, individual hamming loss scores across each sample is added up and the average is taken. If $\hat{Y}$ is the predicted value for the j-th label of a given sample and y is the corresponding true value, and n is the number of classes or labels, then the Hamming loss is defined as:

$$Hamming\ Loss = 1/n \sum_{j=0}^{n-1} 1(\hat{Y} \neq Y) \tag{4.1}$$

where 1(x) is the indicator function.

4.7.2 Subset Accuracy

In multilabel classification, subset accuracy computes the percentage of the set of labels predicted for a sample exactly matching the corresponding set of labels

CHAPTER 5

RESULTS

To evaluate the performance of classifiers, a combination of different feature extraction, selection and classification is used.

5.1 Classification Results

	A	B	C	D	E	F
1	Feature Extraction	Feature Selection	Classification method	Classifier	Hamming Loss	Subset Accuracy
2	TF - BNS	Chi2(K=2000)	Problem Adaptation	Multi label KNN	0.08	0.44
3	TF - BNS	Chi2(K=2000)	Classifier Chain	SVM	0.058	0.572
4	TF - BNS	Chi2(K=2000)	Classifier Chain	SVM(SGD Training)	0.057	0.579
5	TF - BNS	Chi2(K=2000)	Classifier Chain	Light GBM	0.056	0.586
6	TF - BNS	Chi2(K=2000)	Classifier Chain	Gradient Boosting Tree	0.07	0.467
7	TF - BNS	Chi2(K=2000)	Classifier Chain	AdaBoost	0.077	0.443
8	TF - BNS	Chi2(K=2000)	Classifier Chain	XGBoost	0.072	0.468
9	TF - BNS	Chi2(K=2000)	Random K Label Subsets (size=4)	SVM(SGD Training)	0.057	0.574
10	TF - IDF	Chi2(K=2000)	Random K Label Subsets (size=4)	SVM(SGD Training)	0.06	0.534
11	TF - BNS	Chi2(K=2000)	Random K Label Subsets (size=6)	SVM(SGD Training)	0.057	0.583
12	TF - BNS	Chi2(K=2000)	Random K Label Subsets (size=6)	SVM(SGD Training)	0.057	0.585
13	TF - BNS	Model based (Gradient Boosted Tree)	Classifier Chain	SVM	0.057	0.57
14	TF - BNS	Model based (Gradient Boosted Tree)	Classifier Chain	SVM(SGD Training)	0.057	0.579
15	TF - BNS	Model based (Gradient Boosted Tree)	Classifier Chain	Gradient Boosting Tree	0.07	0.467
16	TF - BNS	Chi2(K=4000)	Classifier Chain	SVM	0.056	0.575
17	TF - BNS	Chi2(K=4000)	Classifier Chain	SVM(SGD Training)	0.054	0.594
18	TF - BNS	Chi2(K = 4000)	Classifier Chain	Gradient Boosting Tree	0.07	0.467
19	TF - BNS	Chi2(K=4000)	Random K Label Subsets (size=4)	SVM	0.053	0.592
20	TF - BNS	Chi2(K=4000)	Random K Label Subsets (size=4)	SVM(SGD Training)	0.057	0.575
21	**TF - BNS**	**Chi2(K=4000)**	**Random K Label Subsets (size=6)**	**SVM(SGD Training)**	**0.0546**	**0.596**
22	TF - BNS	Chi2(K=4000)	Classifier Chain	Light GBM	0.055	0.592
23	TF - BNS	Chi2(K=2000)	Random K Label Subsets (size=4)	Light GBM	0.057	0.574
24	TF - BNS	Chi2(K=2000)	Random K Label Subsets (size=6)	Light GBM	0.08	0.484
25	TF - BNS	Chi2(K=2000)	Random K Label Subsets (size=9)	Light GBM	0.178	0.084
26	TF - BNS	Chi2(K=4000)	Random K Label Subsets (size=4)	Light GBM	0.06	0.54
27	TF - BNS	Chi2(K=4000)	Random K Label Subsets (size=6)	Light GBM	0.09	0.45
28	TF - BNS	Model based selection - LighGBM(K=2000)	Classifier Chain	Light GBM	0.055	0.591
29	TF - BNS	Model based selection - LighGBM(K=2000)	Random K Label Subsets (size=4)	Light GBM	0.067	0.526
30	TF - BNS	Model based selection - LighGBM(K=4000)	Classifier Chain	Light GBM	0.055	0.593
31	TF - BNS	Model based selection - LighGBM(K=4000)	Random K Label Subsets (size=4)	Light GBM	0.064	0.542
32	TF - BNS	Model based selection - LighGBM(K=4000)	Random K Label Subsets (size=6)	Light GBM	0.106	0.54
33	TF - BNS	Model based selection - LighGBM(K=4000)	Classifier Chain	Gradient Boosting Tree	0.07	0.46
34	TF - BNS	Model based selection - LighGBM(K=4000)	Classifier Chain	SVM(SGD Training)	0.052	0.597
35	TF - BNS	Model based selection - LighGBM(K=4000)	Classifier Chain	SVM	0.056	0.572

Figure 5.1: Classification Results 1

5.1.1 Inference and observation

- With TF-BNS feature extraction the best subset accuracy was achieved with top 4000 features chosen using chi squared feature selection.

36	*TF - BNS*	*LighGBM(K=2000)*	*Classifier Chain*	*Gradient Boosting Tree*	*0.07*	*0.46*
37	*TF - BNS*	*Model based selection - LighGBM(K=2000)*	*Classifier Chain*	*SVM(SGD Training)*	*0.055*	*0.586*
38	*TF - BNS*	*Model based selection - LighGBM(K=2000)*	*Classifier Chain*	*SVM*	*0.057*	*0.579*
39	TF - IDF	Chi2(K=2000)	Classifier Chain	SVM	0.054	0.59
40	TF - IDF	Chi2(K=2000)	Classifier Chain	SVM(SGD Training)	0.062	0.566
41	TF - IDF	Chi2(K=2000)	Classifier Chain	Gradient Boosting Tree	0.07	0.471
42	TF - IDF	Chi2(K=2000)	Classifier Chain	Light GBM	0.055	0.591
43	TF - IDF	Chi2(K=2000)	Random K Label Subsets (size=4)	Light GBM	0.07	0.5
44	**TF - IDF**	**Chi2(K=4000)**	**Classifier Chain**	**SVM**	**0.051**	**0.601**
45	TF - IDF	Chi2(K=4000)	Classifier Chain	SVM(SGD Training)	0.058	0.58
46	TF - IDF	Chi2(K=4000)	Classifier Chain	Gradient Boosting Tree	0.07	0.469
47	TF - IDF	Chi2(K=4000)	Classifier Chain	Light GBM	0.055	0.593
48	TF - IDF	Chi2(K=4000)	Random K Label Subsets (size=4)	SVM(SGD Training)	0.057	0.577
49	TF - IDF	Chi2(K=4000)	Random K Label Subsets (size=4)	SVM	0.051	0.6
50	**TF - IDF**	**Chi2(K=4000)**	**Random K Label Subsets (size=6)**	**SVM**	**0.051**	**0.606**
51	TF - IDF	Chi2(K=4000)	Random K Label Subsets (size=4)	SVM(SGD Training)	0.057	0.562
52	TF - IDF	Model based selection - LighGBM(K=2000)	Classifier Chain	SVM	0.053	0.58
53	TF - IDF	Model based selection - LighGBM(K=2000)	Classifier Chain	SVM(SGD Training)	0.06	0.569
54	TF - IDF	Model based selection - LighGBM(K=2000)	Classifier Chain	Gradient Boosting Tree	0.07	0.471
55	TF - IDF	Model based selection - LighGBM(K=2000)	Classifier Chain	LighGBM	0.05	0.592
56	TF - IDF	Model based selection - LighGBM(K=2000)	Random K Label Subsets (size=4)	SVM(SGD Training)	0.061	0.552
57	TF - IDF	Model based selection - LighGBM(K=2000)	Random K Label Subsets (size=4)	SVM	0.053	0.593
58	TF - IDF	Model based selection - LighGBM(K=2000)	Random K Label Subsets (size=4)	LighGBM	0.069	0.512
59	TF - IDF	Model based selection - LighGBM(K=2000)	Random K Label Subsets (size=6)	SVM	0.053	0.598
60	TF - IDF	Model based selection - LighGBM(K=2000)	Random K Label Subsets (size=6)	SVM(SGD Training)	0.059	0.565
61	TF - IDF	Model based selection - LighGBM(K=2000)	Random K Label Subsets (size=6)	LighGBM	0.118	0.337

Figure 5.2: Classification Results 2

Using the top 4000 features, a subset accuracy of 59.6% and hamming loss of 0.0546 was achieved with Random K Label Subsets with label size = 6 and SGDClassifier.

- With TF-IDF feature extraction the best subset accuracy was achieved with top 4000 features chosen using chi squared feature selection. Using the top 4000 features, a subset accuracy of 60.6% and hamming loss of 0.0510 was achieved with Random K Label Subsets with label size = 6 and LinearSVC.

- Without feature selection, a major performance problem was noted in the baseline methods. Methods like Random K Label subsets and binary relevance failed to run in a virtual GCP instance even with a 60GB Ram. Classifier chain on a LinearSVC gave a hamming loss of 0.05 and accuracy of 60% without feature selection (190038 features), but the same hamming loss was observed with only 4000

62	TF - IDF	Model based selection - LighGBM(K=4000)	Classifier Chain	SVM	0.051	0.597
63	TF - IDF	Model based selection - LighGBM(K=4000)	Classifier Chain	SVM(SGD Training)	0.058	0.578
64	TF - IDF	Model based selection - LighGBM(K=4000)	Classifier Chain	Gradient Boosting Tree	0.07	0.469
65	TF - IDF	Model based selection - LighGBM(K=4000)	Classifier Chain	LighGBM	0.055	0.592
66	TF - IDF	Model based selection - LighGBM(K=4000)	Random K Label Subsets (size=4)	SVM	0.059	0.555
67	TF - IDF	Model based selection - LighGBM(K=4000)	Random K Label Subsets (size=4)	LighGBM	0.07	0.512
68	**TF - IDF**	**Model based selection - LighGBM(K=4000)**	**Random K Label Subsets (size=6)**	**SVM**	**0.052**	**0.601**
69	TF - IDF	Chi2(K=2000)	Problem Adaptation	Multi label KNN	0.03	0.36

Figure 5.3: Classification Results 3

features which took lesser runtime and memory. 5.1 5.2 and 5.3 show the performance of baseline methods with multi label feature selection.

- Existing literature that compare performance of Multi Label Classification methods on benchmark datasets don't employ any type of feature selection methods[14][15][16]. Majority of the datasets that are used for multilabel text classification in literature are very small both in number of instances and features compared to the dataset that is used in this project. When the baseline methods were applied on these datasets no memory or performance problems were noticed. But with the rise in number of features and instances multi label feature selection is really important to avoid memory and performance bottlenecks.

- Table 5.2 presents hamming loss results for multi label text classification benchmark datasets. The following datasets are relatively smaller both in number of features and records compared to the arxiv dataset used for this project study. (number of features is equal to the total number of distinct words in the dataset.)

Table 5.1: Datasets used

Dataset	*Records*	*Features*
Enron[17]	1702	1054
Reuters-21578	2000	250
Slashdot[18]	3782	1101

Table 5.2: Hamming Loss

Dataset	*BR*	*CC*	*Rakel*	*MLKNN*
enron	0.0049	0.0537	0.0557	0.0699
reuters	0.0793	0.0914	0.0774	0.0714
slashdot	0.041	0.049	0.040	0.089

REFERENCES

[1] Gil Joon-Min Kim, Sang-Woon. Research paper classification systems based on tf-idf and lda schemes. *Human-centric Computing and Information Sciences*, 9:30, 2019.

[2] Iz Beltagy, Kyle Lo, and Arman Cohan. scibert: A pretrained language model for scientific text. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, 2019.

[3] Pedram Amoli and Omid Sh. Scientific documents clustering based on text summarization. *International Journal of Electrical and Computer Engineering*, 5:782–787, 08 2015.

[4] Grigorios Tsoumakas and I. Vlahavas. Random k -labelsets: An ensemble method for multilabel classification. 4701:406–417, 08 2007.

[5] Gang Kou, Pei Yang, Yi Peng, Feng Xiao, Yang Chen, and Fawaz E. Alsaadi. Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods. *Applied Soft Computing*, 86:105836, 2020.

[6] Jesse Read, Bernhard Pfahringer, Geoffrey Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine Learning*, 85:254–269, 08 2009.

[7] Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038 – 2048, 2007.

[8] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[9] Dask Development Team. *Dask: Library for dynamic task scheduling*, 2016.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[11] P. Szymański and T. Kajdanowicz. A scikit-based Python environment for performing multi-label classification. *ArXiv e-prints*, February 2017.

[12] Edward Loper Bird, Steven and Ewan Klein. Natural Language Processing with Python. *O'Reilly Media Inc*, 2009.

[13] Colin B. Clement, Matthew Bierbaum, Kevin P. O'Keeffe, and Alexander Amir Alemi. On the use of arxiv as a dataset. *ArXiv*, abs/1905.00075, 2019.

[14] Ricardo Sousa and João Gama. Multi-label classification from high-speed data streams with adaptive model rules and random rules. *Progress in Artificial Intelligence*, 7, 01 2018.

[15] Passent Elkafrawy, Amr Mausad, and Heba Esmail. Experimental comparison of methods for multi-label classification in different application domains. *International Journal of Computer Applications*, 114:1–9, 03 2015.

[16] Ankit Pal, Muru Selvakumar, and Malaikannan Sankarasubbu. Magnet: Multi-label text classification using attention-based graph neural network. *Proceedings of the 12th International Conference on Agents and Artificial Intelligence*, 2020.

[17] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, *Machine Learning: ECML 2004*, pages 217–226, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[18] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. In Wray Buntine, Marko Grobelnik, Dunja Mladenić, and John Shawe-Taylor, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 254–269, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[19] Newton Spolaôr, Everton Cherman, Maria-Carolina Monard, and Huei Lee. A comparison of multi-label feature selection methods using the problem transformation approach. *Electronic Notes in Theoretical Computer Science*, 292:135–151, 03 2013.

[20] Philipp Probst, Quay Au, Giuseppe Casalicchio, Clemens Stachl, and Bernd Bischl. Multilabel classification with r package mlr. 2017.