Community detection for testing hypothesis of dispersion similarity

Y. Shiva Ganesh and Somu Bhargava

Abstract— **In Computer Science engineering field it is very difficult to get information related to subfields which have specialization areas, since there are too many areas to classify, sometimes these areas can be intersecting. So to effectively get any information about a specialization area of study, we can cluster the research papers which similarities and related to that subfield. So if we want information about specialization area we just have to see the cluster of research papers related to that.**

I. INTRODUCTION

In order to do the clustering on given set of research papers we have to first know what are the similarities between research papers that effect clustering. We can that these are some type of similarities between two research papers, they have intersecting set of authors, they are published in same time period (at some period research papers related to son area will be more published like in past recent years there are more papers published in data mining,machine learning etc), they have intersecting set of citations, they explain about same algorithm or concept, we can tell this by extracting keywords from research papers.

II. DATA SETS

A. Collecting research papers

Suppose we want to get information related to all specialization areas in a subfield of computer science, then we should gather as many research papers related to that subfield, the dataset should be fairly large and most of the specializations related papers should be in the data set.We can get this dataset from web crawling. or by searching through internet.

B. Converting all to same format

Since most research papers are available in PDF formats, we wrote a bash script convertAllPdf2Text.sh that converts PDF file to text file which are easy for processing.

III. PRE-PROCESSING

Pre-Processing is essential for efficient Feature Extraction leading to non-redundant Feature Descriptors. We should do some pre processing to extract keywords from research papers.

- Word tokenization
- Stop words removal

A. Word tokenization

When we are given a sentence list of words are generated by word tokenizor based on delimitors if we have data set of research papers where there are different or complex names or words , then we cannot use general word tokenizor, we should modify the set of delimiters and there may be a need to build a word tokeniszor of our own.

B. Stop words removal

While extracting keywords the general idea is to take those words which occur most in a text file because it may define or give more information about that file. But the common English words which we use at end of sentences will occur more times but they give any information specific to the input file. So we remove all such stop words. The set of stop words English are already there or we can build stop words of our own by adding some computer science technical terms which we use commonly.

IV. FEATURE EXTRACTION

After the pre-processing stage, sufficient amount of redundancy gets removed from the patent content. The words now need to be converted to equivalent feature descriptors to ensure good classification and low enough to ensure that computation performed on them is tractable

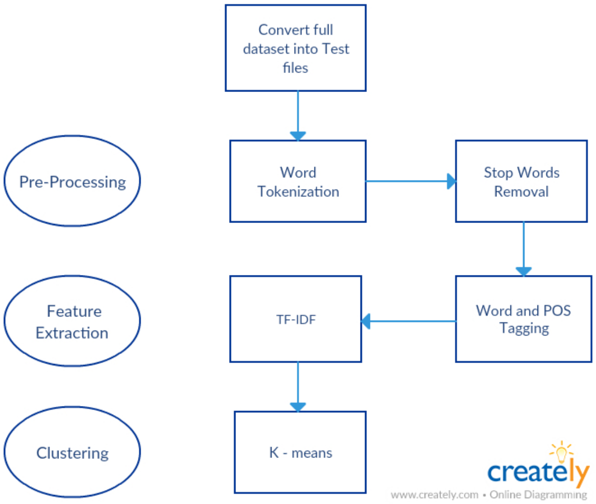
- Word and POS tags
- TF - IDF

A. Word and POS tags

The token itself (in lowercase) was added as a feature. Parts of speech (POS) tagging tags a word with its parts of speech. Most times the keywords are the ones which nouns, verbs other parts speech like pronouns, prepositions etc don't give much information about the input file. POS tags were generated using nltk were also added as a feature. We can now remove other parts of speech words which are of no use.

B. TF - IDF

After the removal of the stop words and useless parts of speech words we are left with set of words which may be keywords. Since the dataset of research papers we collected belong to some subfield or field they contain some words which are commonly used in that field or subfield, these kind of words don't give much information about a research paper and they occur in more frequency in almost all research papers of our dataset. To remove these words we can collect the set of commnly used words in that field but this is a very tiring and huge task, So we use TF - IDF. It means term frequency and inverse document frequency. It takes set of text files as input and gives the set of keywords for every text file. Term frequency refers to no of times a word occured in a file, inverse document frequency means inverse of no of files that occurs in set of files. If it occurs in more files it is of no use to us, it doesn't give specific information. By combining TF and IDF we can get set of keywords of every file in dataset. This set of key words are the features of research paper. We used tf-idf vectorizor from sklearn to get features of every research papers.



V. CLUSTERING

Now that we have features of all research papers we have to cluster them. To do that we should decide in similarity function which tells how similar two research papers are. For this we use cosine similarity, this tells us how much the two sets of keywords of two research papers are intersecting assigns a similarity value to that pair. For clustering we used k-means algorithm. In this algorithm initially there are k means (values) each represent a cluster, if a new member is added to the cluster the mean value of that cluster is updated, it pass once over all members clustering them in one iteration, this way many iterations are done until the clusters are not changing. For implementation of k-means we used nltk. We can as give argument the number of clusters we want.

VI. RESULTS

TABLE I
OBSERVATIONS

No of Clusters	**no of documents in each cluster**
5	78, 52, 5298, 67
8	12, 38, 25, 67, 13, 85, 61, 46
11	21, 18, 31, 34, 29, 31, 11, 47, 32, 46, 47
15	11, 7, 18, 21, 10, 21, 7, 24, 37, 30, 33, 26, 33, 53, 16
18	2, 13, 4, 4, 16, 15, 11, 27, 16, 32, 28, 34, 10, 28, 20, 33, 35, 19
20	3, 4, 17, 6, 13, 5, 10, 11, 20, 10, 11, 30, 3, 11, 33, 52, 6, 38, 26, 38

VII. PRECAUTION

The main precaution we should take is the argument of no of clusters , if we give a small number there won't be clear clustering and ther will be more dissimilarities in a cluster. If we give high argument ,no of clusters will be more and there may even be empty clusters causing the k-means algorithm to stop. So a proper value of no of clusters in some range (range is based on size of dataset) should be given as argument.

VIII. CONCLUSION

Now we can tag each cluster with set of keywords, this way we can get set of research papers related to a specialization in subfield or a subfield in a field.