NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

CZ4079 FINAL YEAR PROJECT

HIERARCHICAL TEXT TOPIC DETECTION

Interim Report

Chen Ziao (U1420681G)

Supervised By –
Prof. Adams Wai Kin Kong
School of Computer Science and Engineering

Table of Contents

INTRODUCTION 3

DATA COLLECTION 3

LITERATURE REVIEW 4

LATENT DIRICHLET ALLOCATION 4
HIERARCHICAL LATENT TREE ANALYSIS 4
FUZZY CLUSTERING 4

METHODOLOGIES 4

DATA PREPROCESSING 4
DIMENSIONALITY REDUCTION 4
AUTOENCODER 5
WORD2VECTOR 5
HIERARCHICAL STRUCTURE GENERATION 5

METHODS COMPARISONS 5

HIERARCHICAL LATENT DIRICHLET ALLOCATION 6
FUZZY CLUSTERING WITH AUTOENCODER 6
FUZZY CLUSTERING WITH WORD2VECTOR 8

NEXT STEP 8

Introduction

This final year project aimed to develop a robust topic detection algorithm to detect topics in a hierarchical structure from raw text documents. The project started with text data collection from online resources. After that, literature review was conducted to understand existing models in topic detection. During literature review, two of the most popular topic detection methods were Ng, Andrew Y's Latent Dirichlet Allocation from University of California, Berkely and Nevin L. Zhang's Hierarchical Latent Tree Analysis from Hong Kong University of Science and Technology. They were both generative statistical model using Bayesian inference to detect topics, which would be further introduced in later section. An interesting fact was that there was almost no paper using traditional clustering. This paper introduced a topic detection method based on Fuzzy C-means Clustering and comparisons between different methods was performed to demonstrate the effectiveness of traditional clustering in topic detection. The last section of this report would briefly describe the tasks for next step.

Data Collection

Data for this project was in raw text format. Python Selenium crawler was implemented to crawl paper data from IEEE website. Due to the large number of papers available, only first 2000 papers ordered by number of citations published from 2001 to 2017 are crawled. All papers were searched using keyword "Computer Science". Due to limitation in computing power, for each paper, only abstract, keywords and title were crawled. Table 1 showed an example of the crawled paper data.

Title	Keywords	Abstract
Algorithm-Dependent Generalization Bounds for Multi-Task Learning	Algorithm design and analysis, Stability analysis…	Often, tasks are collected for multi-task learning (MTL) because they share similar feature structures. Based on this observation, in this…
Bi-Level Semantic Representation Analysis for Multimedia Event Detection	Semantics, Detectors, Training, Multimedia communication…	Multimedia event detection has been one of the major endeavors in video event analysis. A variety of approaches …
Robust Joint Graph Sparse Coding for Unsupervised Spectral Feature Selection with Region Proposal Networks	Manifolds, Computational modeling, Learning systems…	In this paper, we propose a new unsupervised spectral feature selection model by embedding a graph regularizer into the framework of joint sparse regression for …

Table 1. Example of Crawled Data

Literature Review

Latent Dirichlet Allocation

This model was a generative text model which posits each document was generated by a mixture of topics, where the continuous-valued mixture proportions are distributed as a latent Dirichlet random variable. Inference and learning are carried efficiently via variational algorithms. It allowed sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. It assumed that each word in documents are generated by one of the document's topics.

Hierarchical Latent Tree Analysis

HLTA is a novel method for hierarchical topic detection. It modelled document collections using a class of graphical models called hierarchical latent tree models. The variables at the bottom level of the tree represents words in a document. The variables at other levels are binary latent variables. Each latent variable gives a soft partition of the documents. Unlike LDA-based topic models, HLTM did not refer to a document generation process and use word variables instead of token variables.

Fuzzy Clustering

Fuzzy clustering was an extension of K-means clustering. K-means clustering was a hard clustering method where each data point only could belong to one cluster while Fuzzy clustering was a soft clustering method where each data point could belong to multiple clusters at the same time. Fuzziness of the clustering could be determined before running the clustering. However, this method normally applied to low dimension data but not text data because high dimensional data was always very sparse, which would make the clustering centroids tend to converge into a single point.

Methodologies

Data Pre-processing

Since data were all in raw text format, it needed to be pre-processed into a format that could be fed into the topic detection model. Regular natural language processing steps such as stop-words removal, punctuations removal and lemmatization were performed on the raw text data. For this project, lemmatization was chosen instead of stemming because stemming would crudely crop the words, which would the topic detection result hard to interpretation, while lemmatization would preserve the words' form. Using the cleaned text data, tf-idf was done to generate a two-dimensional matrix where each cell contained a tf-idf value for a particular word in a document.

Dimensionality Reduction

Due to the fact that fuzzy clustering did not accept high dimensional data, the tf-idf data matrix needed to perform dimensionality reduction to train the model. The reason why just reducing dimension by selecting some words from the original tf-idf dictionary did not work was that it only represented partial information of the whole document. Therefore, dimensionality reduction was required to perform in a way such that most information was preserved even if much most dimensions were removed. For this project, autoencoder and Word2Vec were chosen.

Autoencoder

An autoencoder was actually a deep neural network where the network was trained to learn the pattern of a data set. Original data set was fed into the input layer and output from the output layer was compared with the original data to using gradient descent learning method to update the network until the error fell under certain threshold. Since tf-idf matrix contained only numeric data, the error function was mean squared error. After the learning was completed, tf-idf data was fed into the network again, but output was obtained from the hidden layer when number of neurons were much less than the input layer. The number of neurons in the hidden layer would be equal to the reduced dimension. Since the result of fuzzy clustering was in fact a collection of centroids of all clusters, all centroids would be fed into the hidden layer and upsampling was performed using the second half of the neural network to recover the original dimension. Finally, based on the recovered tf-idf matrix value, for each cluster, top 20 words with highest tf-idf value would be used to represent this cluster.

Word2Vector

Word2Vector was an algorithm to produce word embedding. It was actually a two layers neural network that were trained to reconstruct linguistic contexts of words. It used a sliding window concept where each word was training using the words nearby inside the window. After model was well trained, each word could be represented as a vector which would be used to perform clustering. After clustering, all centroids were fed into the Word2Vec model to obtain the top 20 most similar words.

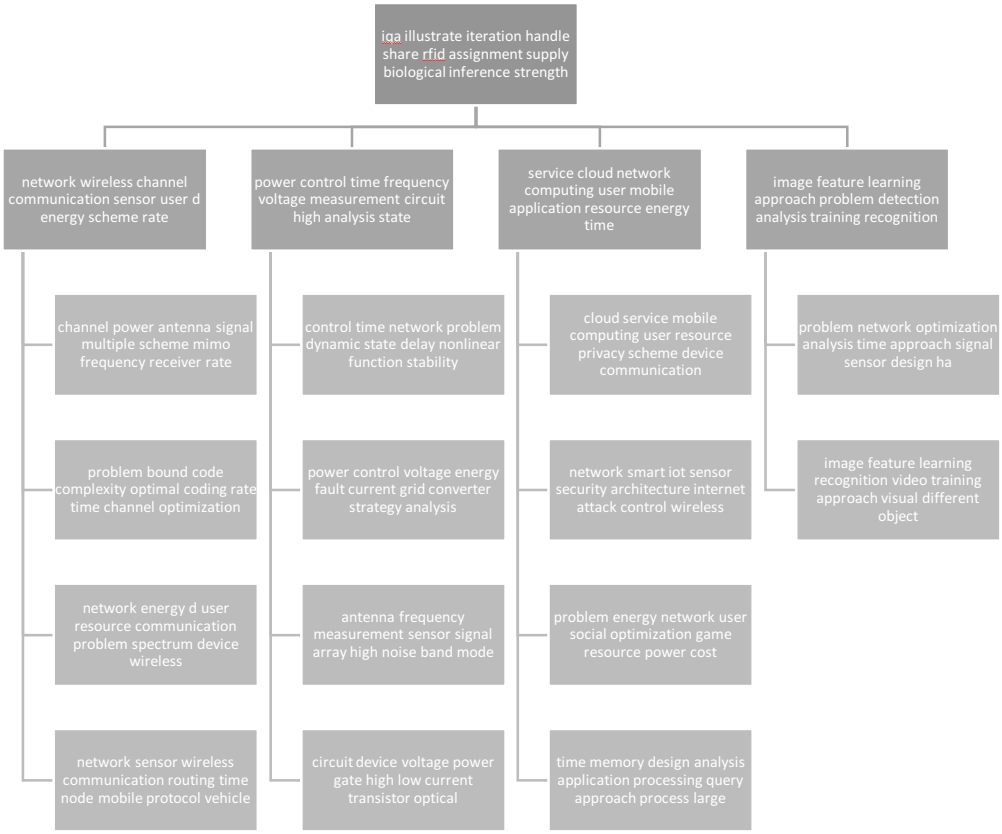
Hierarchical Structure Generation

For fuzzy clustering, it only did a flat clustering without any hierarchy. To generate a hierarchical structure, hierarchical unsupervised fuzzy clustering was used. For each node, there was a weight vector which represented the weight of all points in this cluster. Hence, the children clusters of this cluster were generated using the result of all data points multiplied by the corresponding weight vector. However, the maximum level of the topic tree was pre-determined by the user. For each level, the maximum allowable number of topics also needed to be specified before running clustering. To determine the number of clusters at each level, fuzzy clustering would run multiple times of clustering until it reached the maximum number of clusters. After that, a clustering validity index was used to compared the clustering result of different number of clusters and the best number of clusters would be selected. There were many available clustering validity index and this project used silhouette coefficient and modified partition index.

Methods Comparisons

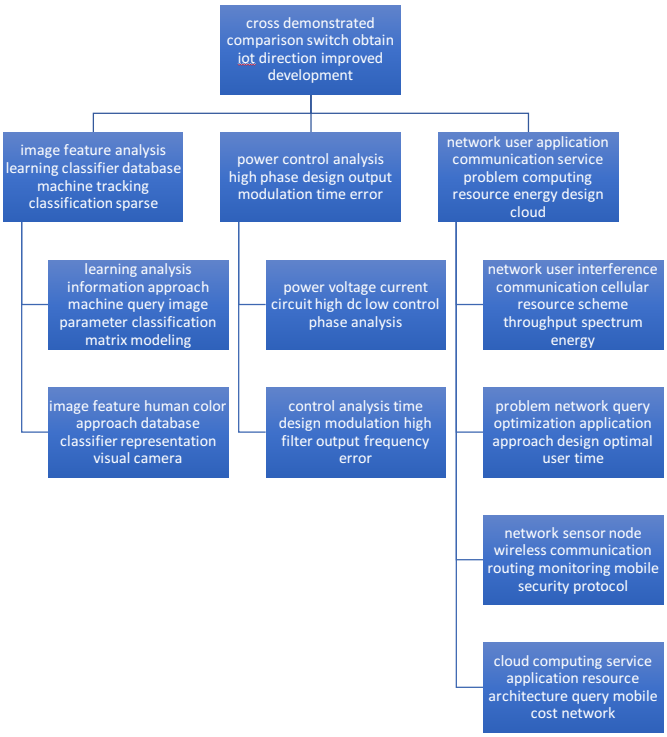
To compare clustering result of different methods, a clustering quality metric called duplication index was designed. For each level in the topic tree, different clusters may share the same representative words, which indicated that the clusters did not separate from each other enough. Therefore, the more number of shared words, the more worse clustering performed. Duplication index used the number of shared words as input and scaled it into a range from 0 to 1. 1 meant completely same clusters while 0 meant perfectly separated clusters. For the following comparison, IEEE 2017 data on keyword “Computer Science” were used and the maximum number of topics at each level was 5. The maximum of the tree was set to be three.

Hierarchical Latent Dirichlet Allocation



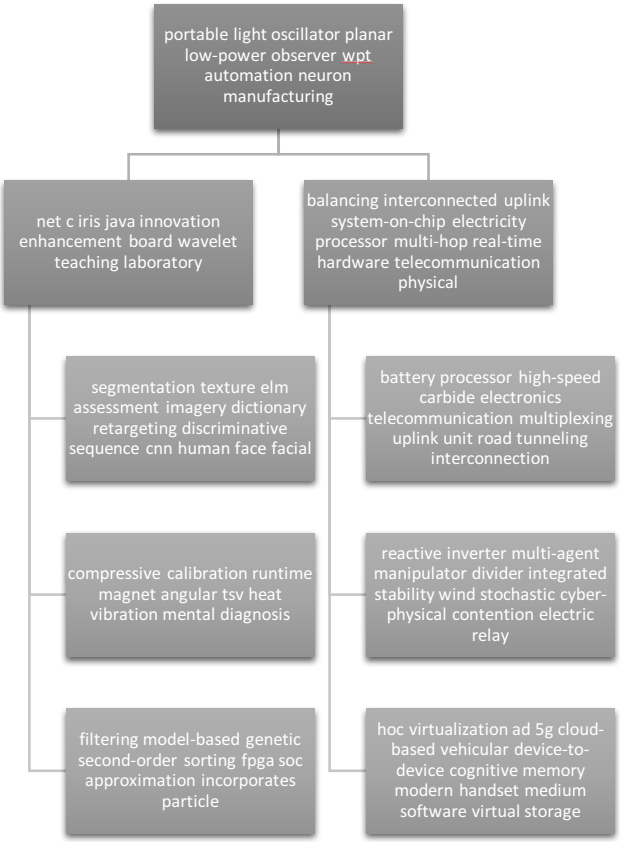
The duplication index for this method was 0.1625 which was also the highest among the three methods. Just from eyeball observation, the cluster did make sense. For example, the second level could be summarized as network, hardware, cloud computing and machine learning. The runtime of this method was moderate.

Fuzzy Clustering with Autoencoder



The duplication index was 0.15 and it was very close to Hierarchical LDA. At second level, the clustering result was similar to Hierarchical LDA except that it only identified three topics which were machine learning, hardware and network and cloud computing. It combined the network and cloud computing into a single topic. However, this method required very long time to train the autoencoder neural network.

Fuzzy Clustering with Word2Vector



The method had the lowest duplication index 0.007 and it was very fast to train the Word2Vector. However, from the topic tree above, it was quite hard to detect the meaning of each topic using the top words. Therefore, this method required to be refined to generate topics that make sense.

Next Step

The next step of this project was to refine the models mentioned above to generate more accurate topic trees. More clustering quality metrics were needed to be designed to fully understand and compare the clustering results. After the best method was chosen, this method would be used to perform topic detection in paper data from 2001 to 2017 and analysis would be done on the result to understand the research topics changing trend and other relevant statistics.