TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

THESIS NO.: 074-MSCSK-013

HETEROGENEOUS GRAPH ATTENTION NETWORK FOR SEMI-SUPERVISED NEWS CLASSIFICATION

by
Sujil Devkota

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN COMPUTER SYSTEM AND KNOWLEDGE ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LALITPUR, NEPAL

August, 2021

HETEROGENEOUS GRAPH ATTENTION NETWORK FOR SEMI-SUPERVISED NEWS CLASSIFICATION

by

Sujil Devkota 074/MSCSKE/013

Thesis Supervisor

Dr. Aman Shakya

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer System and Knowledge Engineering

Department of Electronics and Computer Engineering

Institute of Engineering, Pulchowk Campus

Tribhuvan University

Lalitpur, Nepal

August 2019

COPYRIGHT ©

DECLARATION

I declare that the work hereby submitted for Master of Science in Computer System and Knowledge Engineering (MSCSKE) at IOE, Pulchowk Campus entitled **"Heterogeneous Graph Attention Network for Semi-Supervised News Classification"** is my own work and has not been previously submitted by me at any university for any academic award.

I authorize IOE, Pulchowk Campus to lend this thesis to other institution or individuals for the purpose of scholarly research.

Sujil Devkota
074/MSCSKE/013
Date: September 2021

RECOMMENDATION

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a thesis entitled **"Heterogeneous Graph Attention Network for Semi-Supervised News Classification"**, submitted by **Sujil Devkota** in partial fulfillment of the requirement for the award of the degree of **"Master of Science in Computer System and Knowledge Engineering"**.

.....

Supervisor: Asst. Prof. Dr. Aman Shakya,
Department of Electronics and Computer Engineering,
Institute of Engineering, Tribhuvan University

.....

External Examiner: Assoc. Prof. Dr. Bal Krishna Bal,
Department of Computer Science and Engineering,
School of Engineering, Kathmandu University

.....

Committee Chairperson: Assoc. Prof. Dr. Nanda Bikram Adhikari,
Program Coordinator,
M.Sc. in Computer System and Knowledge Engineering,
Department of Electronics and Computer Engineering,
Institute of Engineering, Tribhuvan University

Date: September 2021

DEPARTMENTAL ACCEPTANCE

The thesis entitled **"Heterogeneous Graph Attention Network for Semi-Supervised News Classification"**, submitted by **Sujil Devkota** in partial fulfillment of the requirement for the award of the degree of **"Master of Science in Computer System and Knowledge Engineering"** has been accepted as a bonafide record of work independently carried out by him in the department.

.....

Prof. Dr. Ram Krishna Maharjan

Head of the Department

Department of Electronics and Computer Engineering,

Pulchowk Campus,

Institute of Engineering,

Tribhuvan University,

Nepal.

ACKNOWLEDGMENT

I have taken the efforts to work on this topic as my thesis. However, without the help of many individuals it would not have been possible. So, I would like to express my gratitude to all of them.

I am very thankful to my supervisor **Dr. Aman Shakya** for his encouragement and precious guidance throughout this thesis work. I am grateful to our program coordinator **Dr. Nanda Bikram Adhikari** for providing a suitable platform for the thesis.

I am highly indebted to **Prof. Dr. Subarna Shakya, Prof. Dr. Shashidhar Ram Joshi**, **Dr. Sanjeeb Prasad Panday, Dr. Dibakar Raj Pant** and **Dr. Basanta Joshi** for their insights and opinions regarding the thesis work.

I would also like to thank our department for providing us the opportunity to work on a thesis in our final semester. Our Head of Department, **Prof. Dr. Ram Krishna Maharjan** and all other faculty members for motivating and supporting us. Lastly, I would also like to thank all of my classmates and my family for all the help and support in this work.

ABSTRACT

Text Classification is one of the important tasks in Natural Language Processing. It involves understanding the semantics and classifying the data into proper class and this method can be further used in many other Natural Language Processing tasks. Since there is a huge amount of data generated every day and one of the major data sources is in text format but finding labeled text data is difficult. So, understanding semantics of such data and analysis has become challenging. Recent trend in graph network has tried to map those raw data to meaningful representations that have a great advantage over less amount of labeled data. The graph network has tried to utilize less amount of labeled data along with unlabeled data and has performed very well in such situations. This work has also explored that technique and has tried to enhance the current work on short text classification. Here, use of heterogeneous graph to represent the raw data has added more semantic to the network as most of the Real-world data are in heterogeneous form. In this work, the raw news data is converted to 3 types of nodes and connection(edges) between them which results in the heterogenous graph. Now the heterogeneous neural network is applied to embed the graph to lower dimension. Also, the dual level attention network was applied that has given more attention to more important nodes and edges further increasing the performance of the model. The application of word embedding using pretrained model has simplified the network, optimizing it's both efficiency and performance. The application of this model has outperformed pervious model in classifying the short news data. In AgNews dataset, the accuracy is 76.3% and in TagMyNews dataset the accuracy is 59.7% that is greater than the previous applied model by more than 4% and 3% respectively. Other visual and comprehensive evaluation also shows that the model performed well with less amount of data.

Keywords

Text Classification, Heterogenous Graph Network, Attention Network, Word Embedding

Table of Content

List of Figures

List of Tables

List of Abbreviations

NLP = Natural Language Processing

SVM = Support Vector Machine

TF-IDF = Term Frequency – Inverse Document Frequency

RNN = Recurrent Neural Network

CNN = Convolution Neural Network

DCNN = Dynamic Convolution Neural Network

GCN = Graph Convolution Network

PTE = Predictive Text Embedding

LSA = Latent Semantic Analysis

LDA = Latent Dirichlet Allocation

SGD = Stochastic Gradient Descent

GNN = Graph Neural Network

GCN = Graph Convolution Network

GAT = Graph Attention Network

HGAT = Heterogeneous Graph Attention Network

Chapter 1

1. Introduction

1.1 Background

Natural Language Processing (NLP) is the branch of computer science and linguistics which is focused on developing a system that can enable computers to understand and communicate using the human language. It focuses on establishing the relation between machine language and human language. Since natural language is very easy to understand for human beings but become challenging for computers to interpret its meaning. Processing such language and understanding its real meaning is sometimes even difficult for human beings since it doesn't depend on only raw text, various other factors like the tone of the language, expression of the speaker have a greater impact on its meaning. But since only the raw text is feed as input in NLP tasks, so it becomes even more challenging to build a system that can handle various forms of ambiguity.

With the introduction of portable and powerful computing devices, the production and collection of data have increased rapidly. Most of such data consist the text which is written in natural language. So, processing and analyzing such data can result in achieving very valuable information and knowledge from it.

Text classification is one of the sub-fields of NLP which deals with understanding the semantics of the text as well as classifying it into proper group. News classification is one of the examples of text classification. News can be classified into different categories based on its content such as "Political News", "Sports News", "Entertainment News" and so on. Category refers to a group that allows easier navigation among articles. This will help users to access the news of their interest in real-time without wasting any time. When it comes to the news it is much difficult to classify as news are continuously appearing that need to be processed and that news could be never-seen-before and could fall in a new category.

1.2 Problem Statement

Even there are large amount of data produced every day it is very difficult to get the meaningful data. Large increase of digital users and digital contents has made the data of any domain to be increased in huge amount. But getting the meaningful labelled data is extremely difficult.

As like other problem in Machine Learning, Text classification problem also requires labelled text data to learn and construct a model. This has created a problem to address how to get information form the unlabeled data and extract meaning from it. For such case semi-supervised learning methods are used which can take advantages of both labeled and unlabeled data.

There is also trade-off on computation of the model when huge amount of data is used to train it. More data makes the model more efficient but in other hand requires high computational power to build the model. So, to address the both case converting the text data to graph network and then learning from it. This makes possible to learn more from less data, utilizing the rich semantic network that the graph can offer and also it can take advantages of the unlabeled data.

1.3 Objectives

1. To convert raw text data to rich semantic heterogeneous graph network.
2. To classify the short news data using semi-supervised HGAT method enhancing the embedding approach.

Chapter 2

2. Literature Review

Various approaches can be used in News Classification such as Rule-based, Decision Trees, Support Vector Machines, Neural Networks, etc. Some of the text classification approaches are:

Traditional Text Classification: Traditional text classification includes methods such as Support Vector Machine (SVM). These methods require feature engineering step for text representation (Drucker et al., 1999). The most commonly used features are BOW and TF-IDF (Blei et al., 2003). Similarly, Naïve Bayes Classifier has also been used to classify the News with comparatively lower accuracy. This is due to the use of TF-IDF which assumes independence between words in the text but in practice words in the corpus are context dependent.

Deep Neural Networks for Text Classification: In deep neural networks texts are represented as embeddings and it was very popular for text classification. Doesn't require feature engineering steps. as RNNs (Liu et al., 2016; Sinha et al., 2018) and CNNs, (Kim, 2014; Shimura et al., 2018) are two deep neural network that have shown their effectiveness in many NLP tasks including text classification. Many different variants on deep neural network were proposed and used over the time. For example, character level CNN (Zhang et al., 2015) which alleviates the sparsity by mining different level of information in the text. Another method purposed was to incorporate entities from knowledge base to enrich semantics (Wang et al., 2017). However, these methods can't capture the semantics properly and it relies heavily on training data. So, lack of training data was key bottleneck in this method.

Semi-supervised Text Classification: Due to the lack of labeled data, this method was formulated to utilize the unlabeled data which are present in huge amount. Since unlabeled data can also provide valuable information and cost of human labeling to produce large labeled data is inefficient. It is basically divided to two category which are Latent variable models (Lu and Zhai, 2008; Chen et al.,2015); and Embedding-based models (Meng et al., 2018). Latent variable models

extend topic model using user provided seed information and embedding based uses seed information to derive embeddings for documents. Example: PTE (Tang et al., 2015) models documents, words and labels with graphs and learns node embeddings for classification.

Since graph can become a very useful tool in semi-supervised learning, recent introduction of graph convolutional networks (GCN) (Kipf and Welling, 2017) have become very popular for text classification. Some of the application of GCN in text classification are: Text GCN (Yao et al., 2019) which models the whole text corpus as document-word graph and applies GCN for classification. However, these models don't focus on attention to capture more important information.

2.1 Graph Neural Network (GNN)

Graph neural networks are extensions of neural networks to structured data encoded as a graph. Originally it was extension of Recurrent Neural Network (RNN) and each node is applied to recurrent layer with local averaging layer. Graph has more dynamic structure and tries to represents the data of real world in more meaningful way. Unlike CNN which has grid structure and fixed neighbor number, in graph neighbor can vary and also the structure can be in any way. Order doesn't matter in graph representation. So, it models the nodes and edges which is basic component of the graph. It has high performance and high interpretability.

Real-world problems can be represented by graphs of different types, ranging from simplest undirected graphs to complex heterogeneous structures. Early graph neural networks worked on undirected graphs and along with it came numerous limitations. Directed graphs have two types of edges, which are easily modeled by using two different weight matrices for each type, from which different adjacency matrices are calculated. Some of the graph types are:

Directed and Undirected Graphs: Directed graph has directed edges which show one way relationship between two nodes. One node having the directed edge to another is related to that node but that connected node is not related back to the main node. Whereas in case of undirected graph if there is edge then both nodes are related to each other.

Homogeneous and Heterogeneous Graphs: Homogeneous graph has same type of node and edges. Example a social network with user friend list can be consider as homogeneous graph, here all user and friend's nodes are same type User and edges between them is friendship. But when there is multiple types of nodes or edges then graph is considered as heterogeneous.

Graphs with Edge Information: Graph can have either information in the edges or no information. No information means just a plain connection with same wights, but when there is information in edge it can be used as strength of the connection (edge). Information can be any like in previous example when two users were friends then date of their friendship can be.

Dynamic Graphs: Another variant of the graph is a dynamic graph, which has a static graph structure and dynamic input signals to capture both kinds of information.

2.2 Graph Convolution Network (GCN):

Graph convolutions Network is a graph neural network with addition of graph convolution layer. It was first applied to text classification in Text GCN (Yao at el., 2018) where the embedding of the nodes was calculated based on the properties of neighboring nodes. With one layer of convolution, it can capture only one level neighbor and with multiple level of convolution it can capture larger neighborhood nodes. There is always trade-off when using the layer of convolution so optimum layer have to be derived.

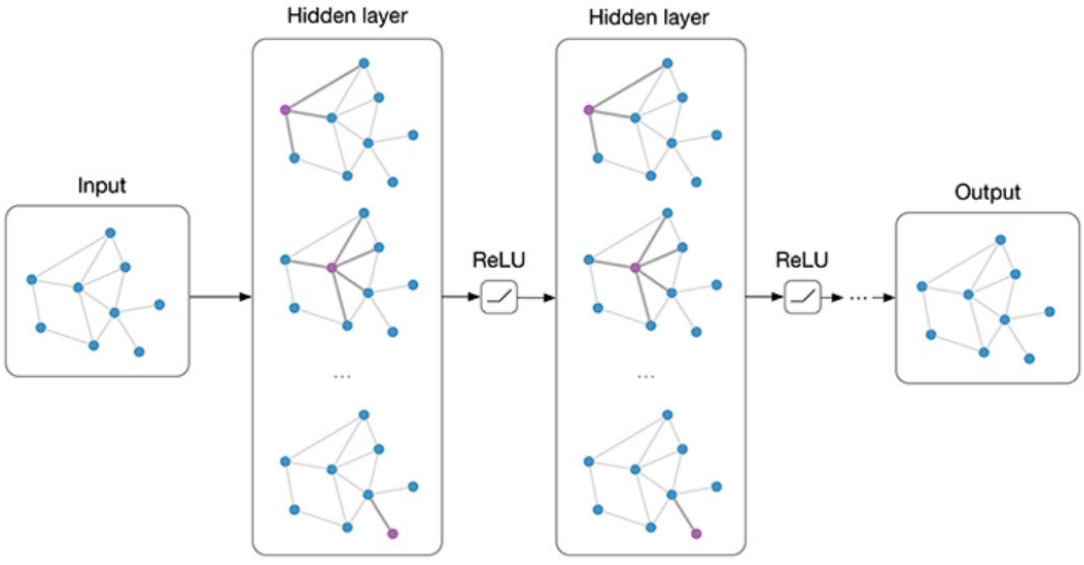


Figure 1: Multi-layer Graph Convolutional Network (GCN) with first-order filters [12]

For 1-layer GCN, node feature matrix

$$L^{(1)} = \rho(\bar{A} X W_0)$$

where,

$\bar{A} = D^{-1/2} A D^{-1/2}$ is the normalized symmetric adjacency matrix

$W \in \mathbb{R}^{m \times k}$ is a weight matrix

ρ is an activation function, e.g., RELU

For higher order,

$$L^{(j+1)} = \rho(\bar{A} L^{(j)} W_j)$$

J denotes the layer number, if j = 0 then $L^{(0)} = X$

Comparison Chart among various Architecture

Title	Semi-supervised Classification with Graph Convolutional Networks (GCN)	Inductive Representation Learning on Large Graphs (GraphSAGE)	Graph Convolutional Networks for Text Classification (Text GCN)
Author	Thomas N. Kipf, Max Welling	William L. Hamilton, Rex Ying, Jure Leskovec	Liang Yao, Chengsheng Mao, Yuan Luo
Date of Pub.	2017	2017	2018
Summary	scales linearly in the number of graphs edges and learns hidden layer representations that encode both local graph structure and features of nodes uses an efficient layer-wise propagation rule that is based on a first-order approximation of spectral convolutions on graphs	It operates by sampling a fixed size neighborhood of each node and then performs a specific aggregator over it. It provides insight into how our approach can learn about local graph structures Future work: to incorporate directed or multi-modal graphs, exploring non-uniform neighborhood sampling functions	It can capture global word co-occurrence information and can utilized limited labelled documents. Simple 2-layer GCN outperforms many state-of-the-art methods on multiple benchmark datasets. Future Work: Improving classification performance by using attention mechanism.

Pros and Cons	- outperforms several recently proposed methods by a significant margin, while being computationally efficient - doesn't support edge features and is limited to undirected graphs	- can learn about local graph structures - effectively trades off performance and runtime by sampling node neighborhoods	- capture global word co-occurrence - more robust, more suitable for less training data than other method - two nodes connecting in a graph may not fall in same class - It gives same score for nodes that are connected to it even
Computational Complexity	Memory requirement grows linearly with size of datasets		O(Pj l=1(rl × rl−1))
Results and Accuracy	Citeseer 70.3 Cora 81.5 Pubmed 79.0 NELL 66.0	Citation GraphSAGE-pool 0.798 0.839 Reddit GraphSAGE-LSTM 0.907 0.954 PPI GraphSAGE-pool 0.502 0.600	20NG - 0.86 ± 0.0009 R8 - 0.9707 ± 0.0010 R52 - 0.9356 ± 0.0018 Ohsumed - 0.6836 ± 0.0056 MR - 0.7674 ± 0.0020

Table 1: Comparison chart for GCN architecture

2.3 Graph Attention Network (GAT):

The Graph Attention Network (GAT) introduces the application of attention mechanisms for graph-structured data. Such a mechanism involves going through all the neighbors of a node to generate the node embedding. The property of propagation through all the neighboring nodes is

somewhat common to that of Graph Convolutional Network but can have better performance. It has been shown to perform well on both inductive and transductive learning for classifying nodes of a graph.
The GAT makes use of masked self-attention units as a basic building block of the network. The term masked refers to incorporating the structure of the graph by computing the attention coefficient only if two nodes are connected. This coefficient is then used to give importance to each neighbor while aggregating their features to compute the feature of a particular node. The multi-head attention can also be applied to make the training more stable.

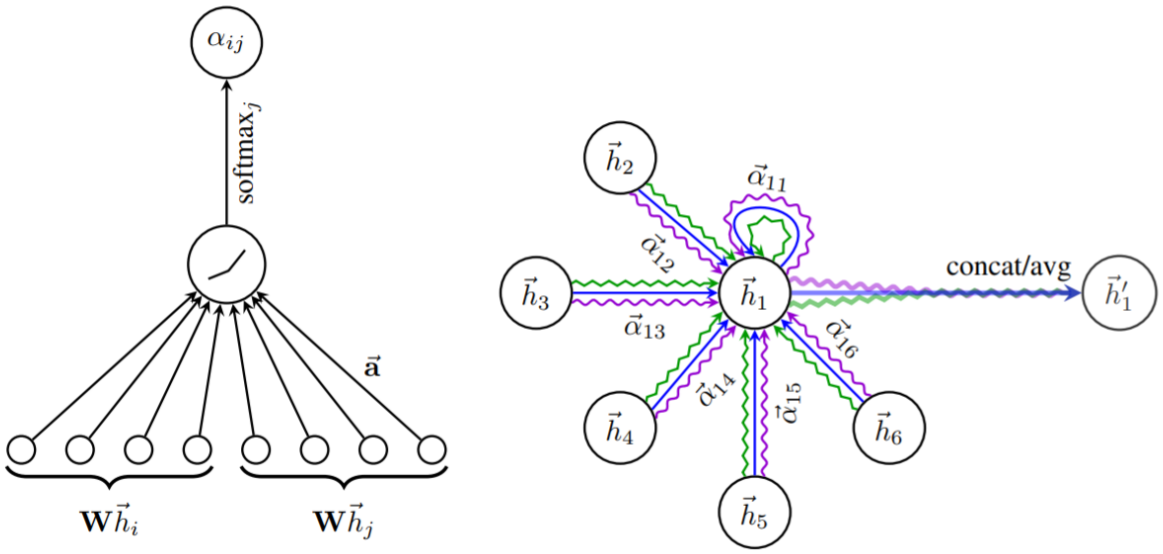


Figure 2: Attention mechanism and illustration of multi head attention with head(k) =3 [1]

GAT has improved and solved issues of related past models. It does not require any heavy computations such as Eigendecomposition. The complexity is comparable to GCN and furthermore, the multi-head attention allows parallel computation as the computation for each head is independent. Attention weights can be visualized and easily interpreted compared to GCN. Also, GAT gives different importance for each neighbor of a node. Similarly, the GAT also solves issues related to the spectral graph neural networks that required knowledge of the structure of the graph prior to the computation. This allows the sharing of computation across the edge of the graph, making inductive learning viable and compatible with directed edges. As all the nodes are considered at once, there is no need to evaluate what nodes to include in the computation and what order of node is important [Hamilton et al 2017]. GAT also greatly reduces the space complexity by using sparse matrices.

However, there are some limitations of GAT. First of all, as the batch size is limited, the GAT cannot deal with a higher batch size, especially for multiple graphs. Secondly, the receptive field

is also limited to the depth of the graph. Finally, the parallel computations might be redundant because of the overlapping of the neighbors.

2.4 Heterogenous Graph Attention Network (HGAT):

Unlike Graph Attention Network (GAT), in addition to attention for homogeneous graphs, Heterogeneous Attention Network (HAN) can work with heterogeneous graph. The most important building block for HAN is meta-path. The idea of meta-paths and attention was developed to deal with heterogeneous graphs, which contain different types of nodes. The heterogeneity is not only limited to nodes but there are also graphs with edge information and multiple types of edges. One of the ways to handle this is to convert it to a bipartite graph, where we replace edges with a node connected to the original nodes. HAN make use of hierarchical information through node-level and semantic-level attention. Other than the ability to collect various types of deep information related to different types of nodes and edges, its advantages are similar to GAT. HAN is also computationally efficient and can be parallelized. The use of different types of attention makes it even more interpretable and easy to visualize. Finally, the parameter sharing enables it to be used as inductive problem solver and be able to generalize to unknown nodes and graphs.

There are many models that make use of attention mechanism such as: AttentionWalks (Abu-El-Haija et al. 2017), GAKE (Feng et al. 2016), GAT (Velickovic et al. 2018), AGNN (Thekumparampil et al. 2018). However, these models do not support heterogeneous graphs. ESim (Jingbo et al., 2016) makes use of heterogeneity and meta-paths but does not involve attention or importance between different meta-paths. Metapath2vec (Yuxiao et al., 2017) is able to address the problem of heterogeneity by using skip-gram connections and random walks but only work with single meta-path and might ignore some important information. Other similar models are out there that deal with heterogeneity but none of them make use of attention mechanism.

The node level attention is followed by semantic level attention to generate the final node embedding. Node level attention is computed for each meta-path. Importance of each meta-path-based neighbors is computed and used to make a weighted combination of all the neighboring node features in that specific meta-path to generate meta-path specific embedding of the particular node. For each meta-path, such embedding is obtained and then aggregated by assigning different importance to different meta-path. This is the semantic-level attention. The output of this semantic-level attention is the final node embedding and can be then connected to a neural network for solving the specific task.

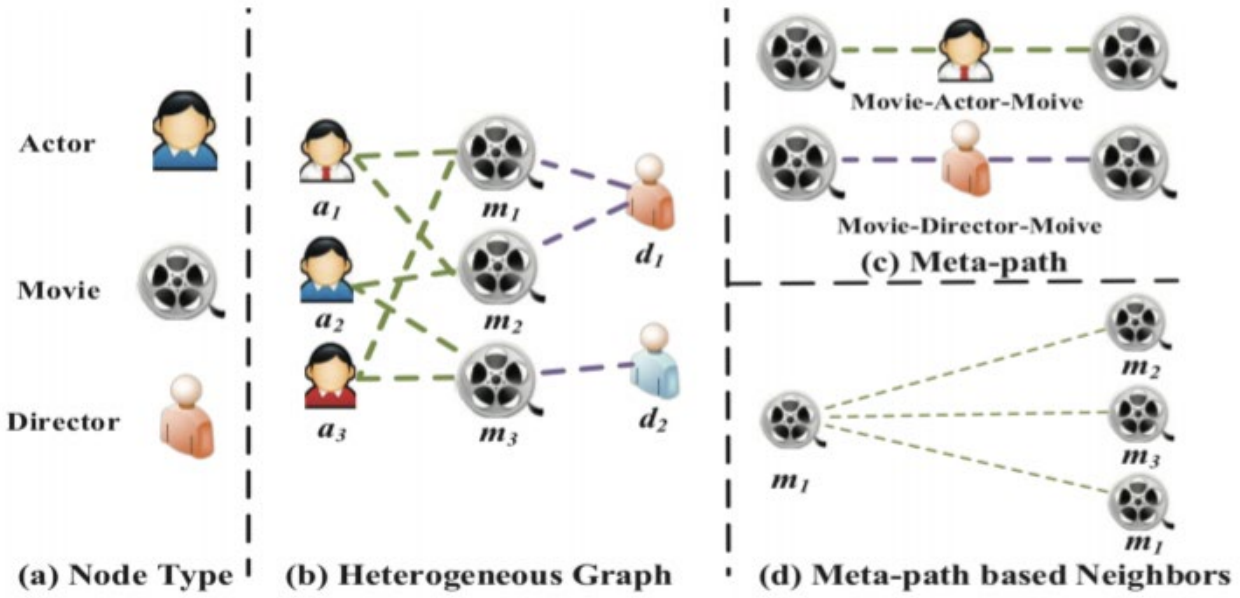


Figure 3: Example of heterogenous graph (IMDB). (a) 3 types of nodes (actor, movie, director). (b) 3 types of nodes & 2 types of connections. (c) 2 meta-paths (iMovie-Actor- Movie & Movie-Director-Movie) [2]

A heterogeneous graph, can be denoted as $GG = (VV, EE)$ where VV is node set and EE is edge set. The node type mapping function can be represented as $\phi\phi$: $VV \rightarrow AA$ and a link type mapping function as $\psi\psi$: $EE \rightarrow RR$. AA and RR denotes the sets of predefined node types and edge types, such as $|AA| + |RR| > 2$.

Chapter 3

3. Methodology

3.1 Block Diagram

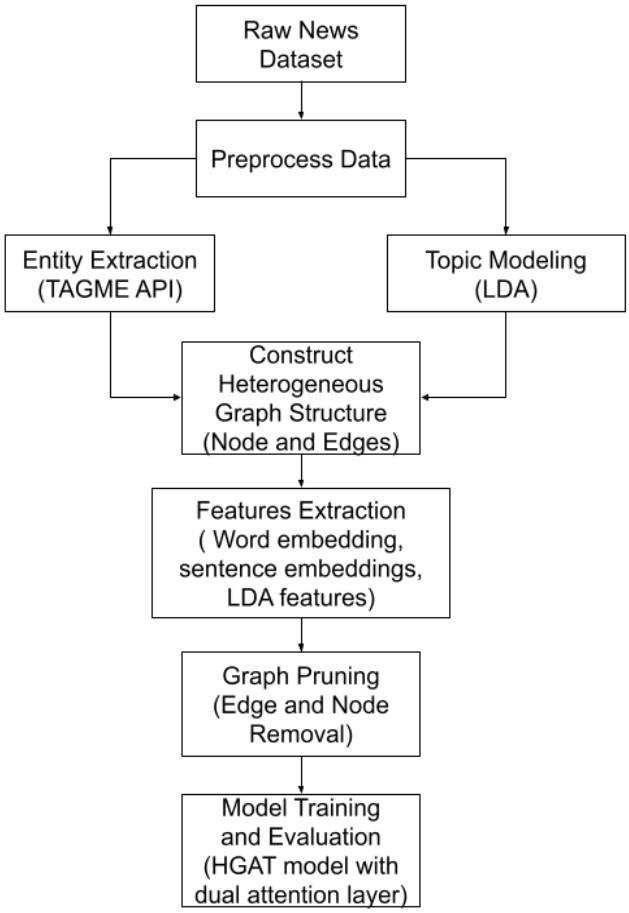


Figure 4: Overall Block Diagram

3.2 Data Collections

AGNews Dataset is used in this work. About Base AGNews Dataset: Original AGNews datasets contain 4,96,835 categorized news articles from more than 2000 news sources of more than 1 year. Version 3, Updated on 09/09/2015.
Derived AGNews Dataset: It is derived from the above main base AGnews dataset. The 4 largest classes from the corpus are chosen to construct the dataset. Training datasets = 120,000 and testing datasets = 7,600. Each class has 30,000 training samples and 1,900 testing samples. Dataset consists of 3 columns: Class, title, description.

Our current dataset is again reduced from the derived AGNews Dataset. It consists of 6,000 data which was sampled randomly where there are 4 categories and each category is evenly distributed. Each category consists of 1,500 news. The 4 categories are: World - 1, Sports - 2, Business - 3, Sci/Tech - 4. (1,2,3,4 are index used to represent those category)

Another dataset used is TagmeNews dataset which consists 200k news headlines from year 2012 to 2018. It is obtained from HuffPost and consists of 41 categories from which only 5 categories are selected which are Politics, Food, Sports, Crime and Science (1,2,3,4,5 are the index used to represent them respectively). Each category is evenly distributed and consists 1,200 news which is randomly selected from the main dataset.

3.3 Data Preparation

The input data should be cleaned and preprocessed before analyzing or applying the model to it. This increases the performance of the system as well as its accuracy to great extent. The various common steps that are used in different stages are:

- Tokenization: Process of separating words from the given text. It divides the text into the smallest unit called tokens.

- Lowercasing: All words are lowercase to reduce the confusion that might treat the same word as different words due to differences in letter case.
- Removal of stop words: Removing the less meaningful words can help in increasing the performance and accuracy. A predefined stop word list file is used for this purpose which consists of 401 words.
- Removal of the special symbol: Various types of symbols that don't have a literal meaning can be removed.
- Lemmatization: It is the process to convert inflected words to their common base word. This step helps to treat the infected and base word as the same.
- Using underscore in place of space: For any single entity which consists of two words separated by space then the space between the word is replaced by an underscore to denote the same entity. Ex: new york is made new_york.

3.4 Graph Construction:

The final data fed to the model is in the form of a graph. So, in the data preparation step, the raw input data is converted into a heterogeneous graph structure. Which involves the following steps:

3.4.1 Entity Extraction:

The first step is to extract the key entities that are present in the documents so that a graph will have an entity node which is connected to the main document node. The key entity of one document appearing in another document means that two documents are somewhat similar to each other. So, adding an entity as a node and forming the edges between the document and entity (DvsE) and also with entity and entity (EvsE) both relations contribute to the rich semantic information in the graph.

For this case, TAGME API is used to get the list of entities in a document. TAGME is a tool

that identifies meaningful substrings or entities in an unstructured text and links each of them to an appropriate Wikipedia page. Its RESTful API is used to annotate the document. This is particularly good for short texts. TAGME performs entity linking in a pipeline of three steps: Parsing, disambiguation, pruning. Entity linking is the task of annotating an input text with entities from a reference knowledge base. Annotation outputs the reference to a Wikipedia page, entity, for a substring of an input text, spot. The entity also represents the meaning of the spot in the context in which the spot is present in the document.

Response from TAGME API:
Text input response->[{entity1}, {entity2}, ...]

Ex: entity1 format:

"spot": "manufacturer",	word in input text (content1)
"start": 12,	position where the word start in input text
"link_probability": 0.0179.,	link(m)/freq(m) in wikipedia
"rho": 0.278..,	measure of goodness with other entities in the input text
"end": 24,	position where the word ends in the input text
"id": 39388,	unique identifier which represents this word
"title": "Manufacturing"	wikipedia base title for this word

link_probabilty = link(m) / freq(m)
link(m) -> total no. of times the entity is mentioned in wikipedia text
freq(m) -> total no. of times entity is mentioned as link in wikipedia

3.4.2 Topic Modeling:

Topics represent the abstract topic that occur in collection of documents. Here the input news text is used to construct the topics. For representing the topic node in the graph topics are extracted using topic modeling through LDA (Latent Dirichlet Allocation). Each topic T_i is represented as

$$T_i = (\theta_1, \theta_2 \ldots\ldots, \theta_w)$$

where, i is the number of topics

θ_w represents keywords and w is the vocabulary size

Here each T_i will form a node and each document is assigned with P topics that have the highest probabilities. To generate topics from the dataset the LDA algorithm is used.

LDA's approach of topic modeling considers each document as a collection of topics in a certain proportion. And each topic as a collection of keywords, again, in a certain proportion. Contrast with like, k-means, where each entity can only belong to one cluster (hard-clustering). LDA allows for 'fuzzy' memberships (soft-clustering). Here one document can have multiple topics. LDA can automatically discover topics based on keywords. A topic is just a representation of a collection of dominant keywords.

Constructing LDA model starts with data preprocessing the above methods explained for data preprocessing are also used here lowercasing, removing stop words, lemmatization, less character word removal. For computing the probability of each word TFIDF approach is used. Term Frequency–Inverse Document Frequency (TFIDF), is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

For a term i in document j

$w_{i,j} = tf_{i,j} * \log(\frac{N}{df_i})$

$tf_{i,j}$= Number of occurrences of i in j

df_i= Number of documents containing i

N = Total no. of documents

The TFIDF representation of all documents is passed to the LDA algorithm which first randomly picks the document into some topic and then keeps updating with each new document addition in each iteration. The number of topics to which the document belong should be passed initially. Topic number from 10 to 20 was tested then coherence score was determined to know which no. of topics can represents the documents more accurately. Also,

the value of alpha and beta were also changed. Tunned Hyperparameters: No. of topics (k), Document-Topic Density (α) and Word-Topic Density (β)

3.4.3 Node and edge construction:

This step involves construction of a graph network. It can be referred to as Heterogeneous Information Network (HIN). The graph consists of three types of nodes which are obtained by processing the document text. Entity Extraction step gives the entity node, another is the document itself and another is constructing the topics using topic modelling. Since all nodes have different data so the graph is considered as a Heterogeneous graph.

Node type: Document - Ⓓ, Topic - Ⓣ, Entity - Ⓔ

Edge type: Ⓓ↔Ⓣ, Ⓓ↔Ⓔ, Ⓔ↔Ⓔ

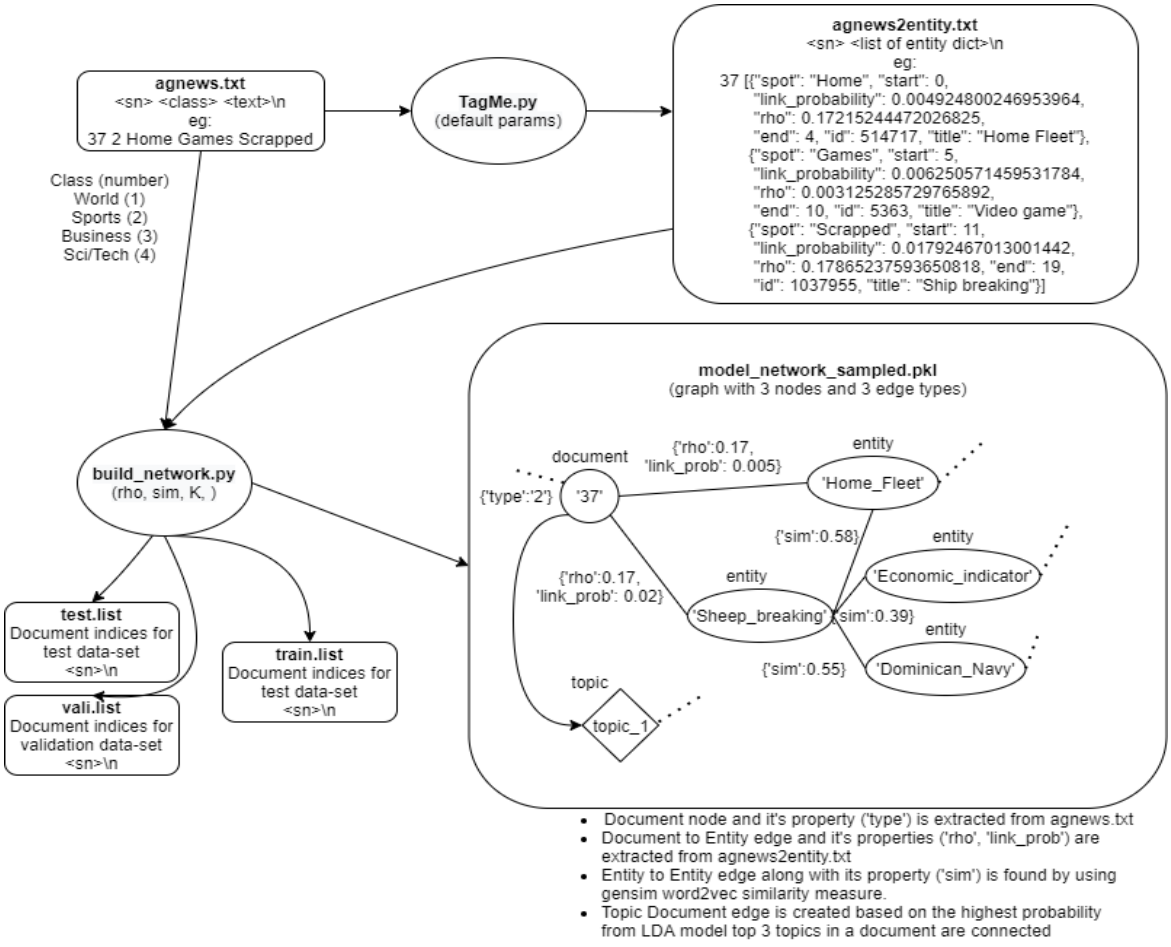


Figure 5: Overall Graph Build Process

Rich Semantics formation in Graph:

How documents are considered close to each other (two documents are linked in following formations with each other). These relationships are also called meta paths.

- Ⓓ↔Ⓣ↔Ⓓ
- Ⓓ↔Ⓔ↔Ⓓ
- Ⓓ↔Ⓔ↔Ⓔ↔Ⓓ

3.4.4 Features Representation:

Graph consists of the three nodes Document(D), Entity(E), and Topic(T). In this step, features are added to each node so that it can be used in training the model. For every 3 nodes, 3 different approaches are done to represent the features.

Entity(E) features using word embeddings:

To represent the entity node, each entity text is converted to word embedding using the google news pre-trained word2vec model. This word2vec model consists of 3 million words and phrases that are pre-trained using 100 billion words from the google news dataset. The size of the model is 3.3 GB. This model gives the entity text in vector form. The dimension of the vector is 300 which is in the compressed form than the traditional TFIDF. This gives us advantages over computation and greatly reduces the training time. Since the dimension is very low then the TFIDF representation, more information is compressed inside the vector.

Document(D) features using sentence embeddings:

To represent the document node features, the text of the document is used to construct a sentence embedding. First the text is passed through basic pre-processing involving lowercasing, removing stopwords, replacing entity title to the entity found from entity extraction. Now each tokenized word is passed to the pre-trained word2vec model which gives a low dimensional feature of each word. Then the sentence embedding is computed using the

centroid method where the sum of each word embeddings in a sentence is weighted by their tf-idf score and divided by the sum of these tf-idf scores. [18]

Topic(T) Features obtained from LDA:

Since LDA is used to obtain the topic node, the LDA vector representation of each topic node is used as its features. So, the LDA model trained in the previous step to construct the topic node gives the features for this node. Analysis and training of LDA model is done using the genism library. The optimum number of topics is found by analyzing the model through coherence score and PyLDvis plot. For this case 15 topics are used to represents the news text. So, 15 topic nodes are formed in the graph. Also, each document node is connected with 2 topics with highest probability.

3.4.5 Graph Cleaning and Pruning:

This step involves the cleaning of the graph before it is sent to the HGAT model. It involves rechecking the parameters that satisfy the edges or node to be in the graph. Some of the parameters that were used in graph construction are: The similarity between the two entities should be more than 0.5 which is given by word2vec similarity function. rho score and link probability of each entity node must be more than 0.3 and 0.75 respectively. If any entity node exists without features, then such entity is removed from the graph. Also, the outlier node with no connection is removed.

The reduced graph is now split to train, test, and validation set to prepare the data for training the model. Also, the graph node adjacency matrix is constructed which is useful for passing the graph to the model. The Adjacency matrix represents the connection between nodes that exist or not, in the matrix form.

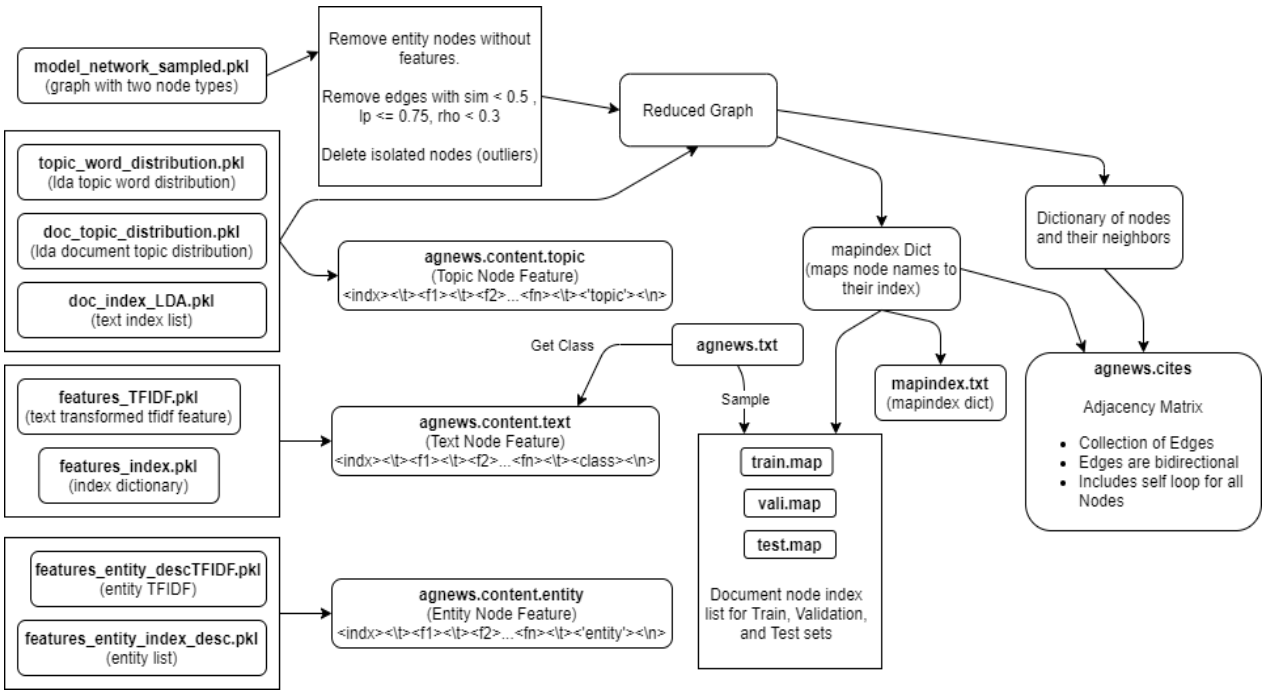


Figure 6: Overall Graph Preprocessing Process

3.5 Heterogeneous Graph Attention Network (HGAT) Model:

3.5.1 Node Embeddings

Node embeddings is the process of computing the node features so that it represents its information about how it lies in the graph. Embedding each node features, converts the graph to low dimensional space and also preserves its structure and property. Some of the embedding methods used in graphs are random walk, deepwalk, Node2Vec etc. these all methods are only used in homogeneous graphs.

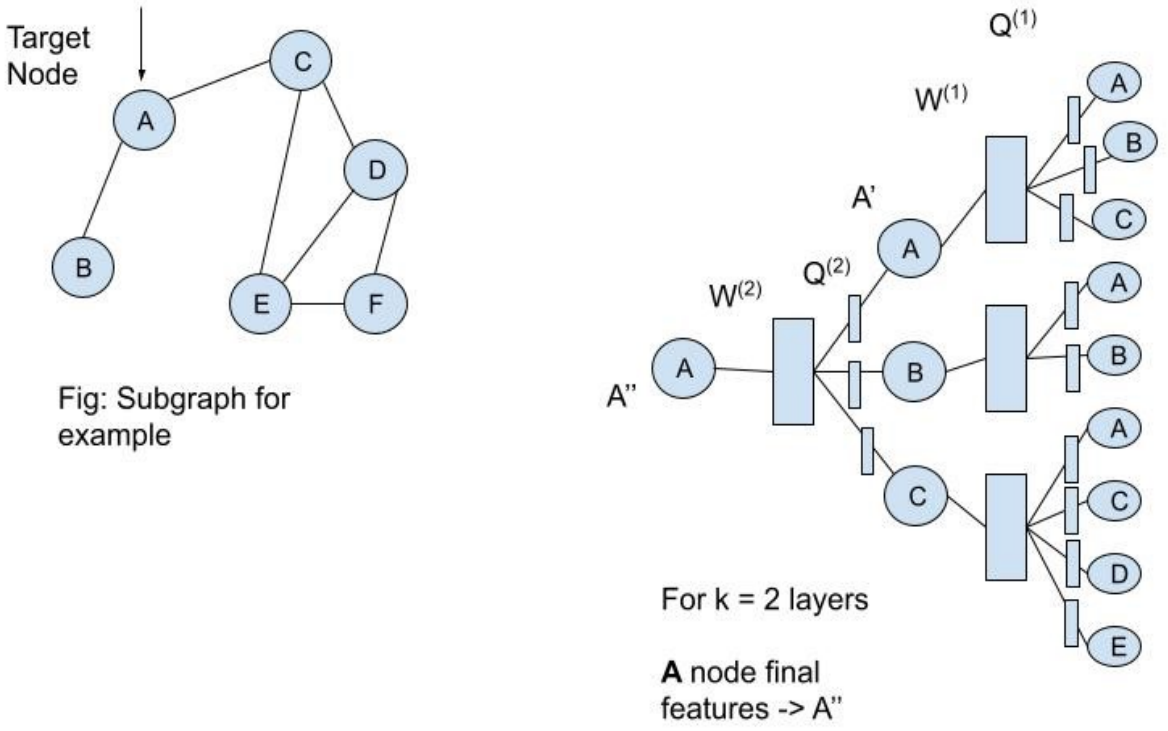


Figure 7: Node embedding in a homogenous graph

For heterogeneous graphs also various types of node embeddings are available such as ESIm, HERec, PME, metagraph2vec, HEER etc. but these methods don't consider about attention while embedding the node.

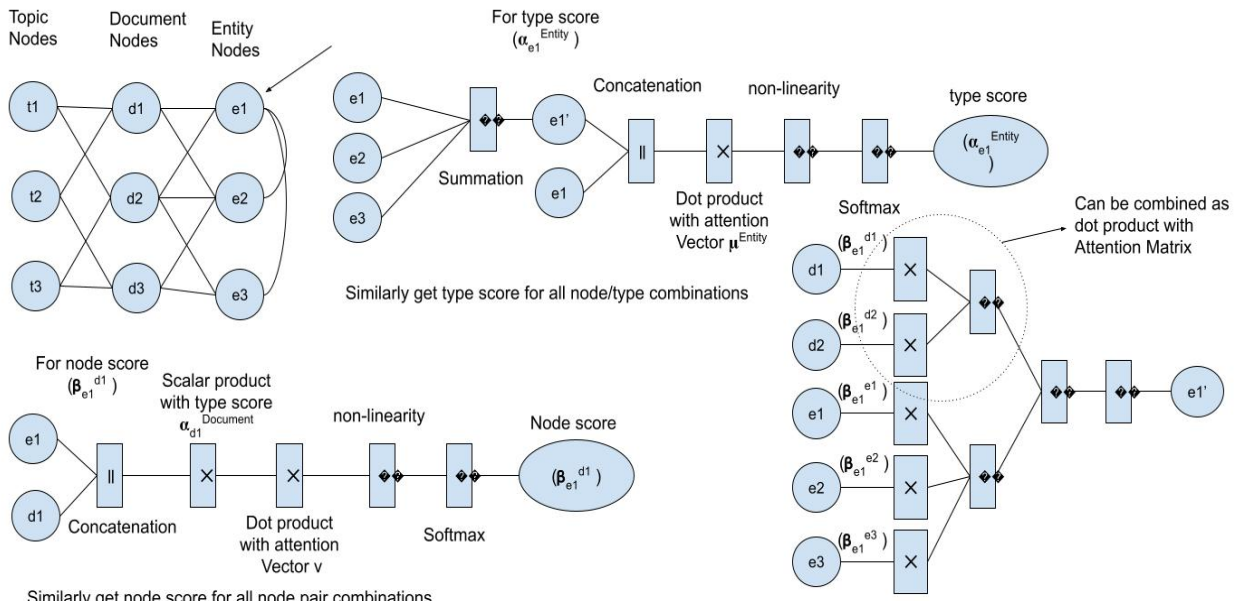


Figure 8: Overall HGAT model architecture

So, Heterogeneous graph embedding method is applied to embed the features of each node. It considers both the heterogeneity and its importance in the graph by assigning attention masks to each node. Attention is the mechanism to find the important information in a model and give priority to it. The information can be a word in a text or it can also be a node in a graph. Attention basically gives more importance to the specific information that has more meaning. Here in the case of graph networks, the attention role is to give importance to each node based on the attention score. Attention is basically applied to two levels: node level and type level.

3.5.2 Node Level Attention

It gives the attention score for each node implying how important that node is to the target node. To apply the attention mechanism to the node multi-head self-attention method is used.

Multi-Head Self-Attention: The basic mathematical representation of self-attention

$$Attention(Q, K, V) = softmax(\frac{Q\,\boldsymbol{K^T}}{\sqrt{n}})\mathrm{V}$$

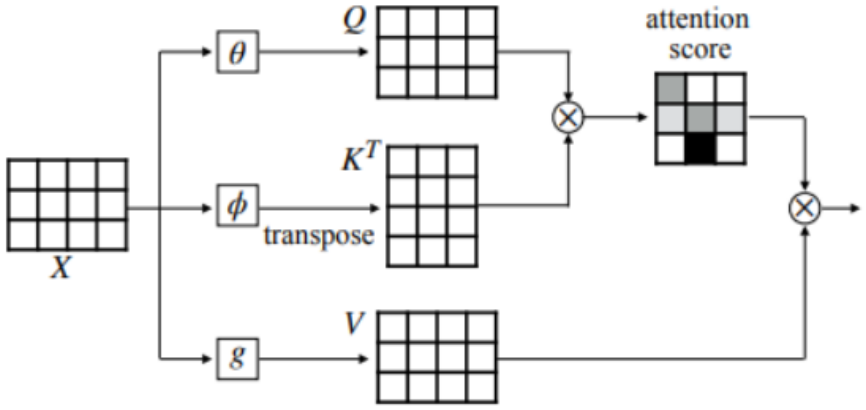


Figure 9: Self Attention Mask Calculation [18]

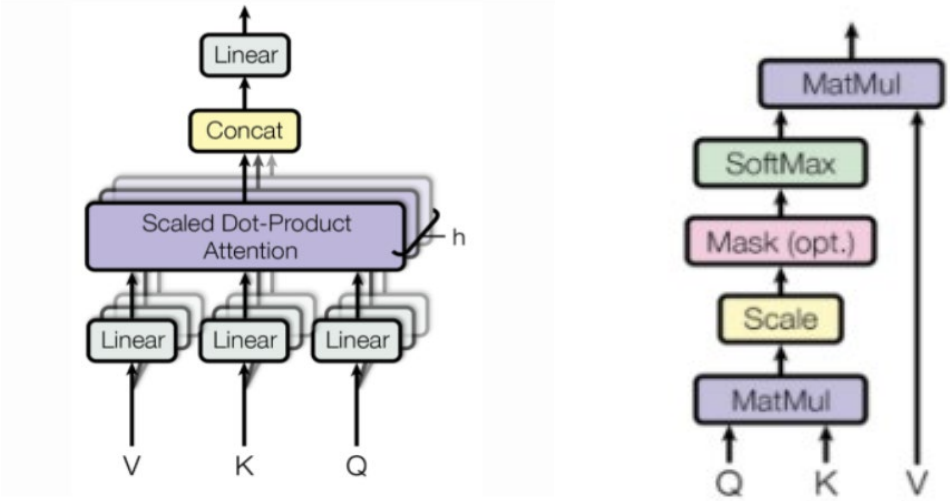


Figure 10: Multi-head Self Attention from paper [5]

Algorithm for computing self-attention

1. Prepare inputs
2. Initialize weights
3. Derive key, query, and value
4. Calculate attention scores for Input

5. Calculate SoftMax
6. Multiply scores with values
7. Sum weighted values to get Output
8. Repeat steps 4–7 for each input

According to the paper, "multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this." [5]

MultiHead (Q, K, V) = [$head_1$ || $head_2$ || … || $head_h$] W_O
where h = no. of head used

$head_i$ = Attention (QW_{Qi}, KW_{Ki}, VW_{Vi})
where W_{Qi}, W_{Ki}, W_{Vi}, and W_O are parameter matrices to be learned.

3.5.3 Type Level Attention

In this level the attention score is based on the type of the node. Which type of node to give more attention is determined. For example, when computing the features of a document node, document type can be considered more important than other types of nodes. As in this method it was observed that for document nodes, entity type has more attention than the topic node since topic is little general type of representation of document.

$$h_T = \sum_{v'} \tilde{A}_{vv'} h_{v'} \qquad (3.1)$$

Here, $h_{v'}$ is the sum of neighboring nodes features, and v' represents same type T nodes.

$$\alpha_T = \sigma(\mu_T^T . [h_{v'} || h_T]) \qquad (3.2)$$

Here, $\mu_T^T$is the attention vector for the type T, || is the concatenation of the current node features with neighbor node features obtained from (3.1). $\sigma(.)$ denotes activation function.

Then the attention vector is passed through SoftMax layer

$$\alpha_T = \frac{exp(\alpha_T)}{\sum_{\pi \epsilon T} exp(\alpha_\pi)} \qquad (3.3)$$

3.5.4 Transductive Learning

In transductive learning the specific data are predicted given the specific data from the domain. It is different from the inductive learning which derives the function from the given data. Transductive learning is also referred to as instance-based learning or class-based learning. In this approach also we use transduction learning for semi supervised classification. Here the whole graph is passed to the model, test data are also passed but their labels are not used. It treats test data as unlabeled data and learns the structure of the graph from it also.

Inductive learning can also be done, for that separate graph to be constructed for train and test and should be passed separately.

Adam Optimizer:

For optimization Adam optimizer is used. It is an optimized version of stochastic gradient descent and has been popularly used in deep learning algorithms. It is the combination of two optimizers: adagrad and rmsprop. Adagrad works well with sparse gradients and rmsprop works well with non-stationary settings. Main idea of Adam is to maintain exponential moving averages of gradients and its square.

Update is proportional to $\frac{average\ gradient}{\sqrt{(average\ squared\ gradient)}}$

Chapter 4

4. Result and Analysis

While analyzing and testing the model lot of hyperparameters are used in different part of the methods. As the method start with construction of graph, while constructing the graph hyperparameters used were:

For entity node, entities with lower value rho, obtained from entity extraction method are discarded.
Rho score > 0.1

For topic node, topic model is constructed using LDA. No. of topic that classify text more meaningfully into each soft cluster is determine through various analysis process: Observation of each topic keywords, coherence score, plotting topics to lower dimension using PyLDvis. Optimum determined values were:
Topic number (K) = 15
Document-Topic Density (α) = 0.1
Word-Topic Density (β) = 0.1
For TagmyNews the Topic number(K) = 17 gives more effective topic distribution

Entity-Entity edge construction, two entities having more than 0.5 similarity score are connected together since they are considered close to each other. Also, one entity is at least connected to 10 other entities even the similarity score is less. (Minimum case)
Similarity score > 0.5
Min entity link > 10

Document-Topic edge construction, each document text is linked with all topics with some probability score but the topic which are highest probability are consider to fall in that document.
Top topic with highest probability (P) = 2

So, each document node is connected to 2 other topic nodes.

HGAT model hyperparameters:

Learning Rate (LR) = 0.005

Dropout Rate (DP) = 0.95

Weight Decay (WD) = 5e - 8

Epoch = 300

Hidden Unit Dimension = 512

4.1 Accuracy and Loss Curve:

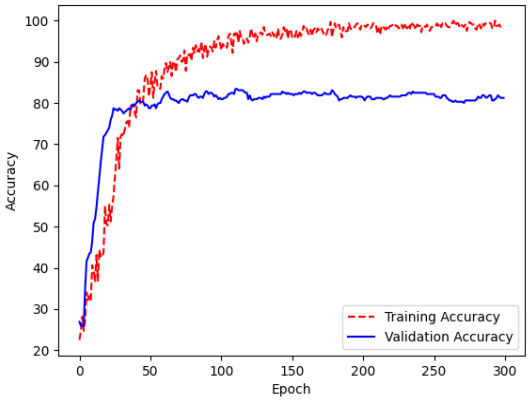


Figure 11: Accuracy Curve (Agnews)

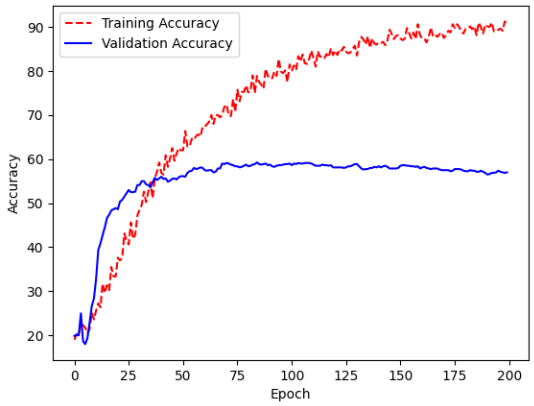


Figure 12: Accuracy Curve (Tagmynews)

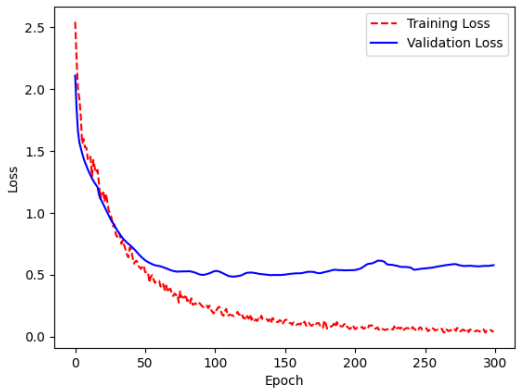


Figure 13: Loss Curve (Agnews)

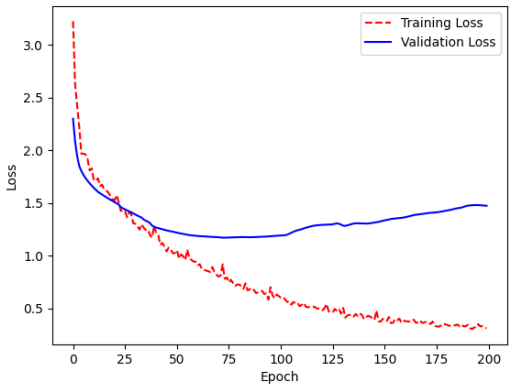


Figure 14: Loss Curve (Tagmynews)

Accuracy and loss curve shows how the model is training and improving in each epoch. It helps to understand how accuracy and loss is varying over training and validation data set.

4.2 Confusion Matrix

Confusion matrix is a table which helps to visualize the performance of a classification model on a set of test data for which the true values are known. It comprises of True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) values which help to derive the following evaluation parameters:

- **Precision**: It is the ratio of true positive by total predicted positive.

  Precision (P) = TP / (TP + FP)
- **Recall**: It is the ratio of true positive by total actual positive.

  Recall (R) = TP / (TP + FN)
- **F1-score**: It is the harmonic mean of Precision and Recall. It combines both factors give more accurate info of the model.

  F1-score = 2 × Precision × Recall / (Precision + Recall)

4.3 Comparison table

Datasets	Document Node	Entity Node	Topic Node	Training Nodes	Validation Nodes	Test Node
AGNews	6000	10298	15	160	160	5680
TagmyNews	6000	6898	17	800	800	4400

Table 2: Dataset Statistics

Metrices / Model	Accuracy	Precision	Recall	F1 Score
HGAT base paper (Agnews)	72.10	-	-	-
HGAT* (Agnews)	**76.30**	**76.4**	**76.3**	**76.3**
HGAT paper method (TagMyNews)	56.75	56.8	56.9	56.7
HGAT* (TagMyNews)	**59.57**	**59.9**	**59.6**	**59.5**

HGAT* applied in this research

Table 3: Comparison and performance of the model

Chapter 5

5. Conclusion and Future Work

5.1 Conclusion

Hence, the heterogeneous graph attention network was applied to the news classification problem where it was able to perform well under very less labeled data, with less text. HGAT has shown its efficiency in short text classification in a semi supervised way. Transductive learning was done to utilize the unlabeled data also in the training process. This network is updated over the GCN to fit the heterogeneity combined with a dual level of attention. In this work the previous work on HGAT was optimized by improving the features embedding methods and also improvement in the attention network.

In this work use of word embedding to embed the features of each entity node and use of centroid method sentence embeddings to embed features of each document node, both reduces the dimensionality space as well as increase the semantic meaning of the text. It also shows an increase in computational efficiency and also update of attention mechanism, use of self-attention network provides a new way to capture more attention perspective over the network.

The method update shows an increased accuracy of 76.30% on classification of AgNews datasets where base paper only shows 72.10%. As well as another dataset, TagMyNews dataset was also used to test both model where the base paper method gives 56.75% accuracy and the updated model in this work result in 59.57% accuracy resulting increased performance in both dataset by around 4% and 3% increment. Agnews has 4 different class of news to classify while TagMynews consists 5 different class of news both having same amount of data.

5.2 Limitations

Limitation in this work is about high specs computer is needed to run the model smoothly. Minimum required RAM is about 12 GB for the dataset used here. If the dataset is increased, then

the memory required also increases greatly. Model needs to be retrained in case of topic modeling when the new type of data is fed. Also, it is limited to classify news to known category only.

5.3 Future Works

There are lot of methods combined in this work, from entity extraction, topic modeling, text preprocessing, text embeddings, attention network etc. So, individual method can be improved and how it will affect the overall model can be studied. The topic model in this work is not dynamic so recent topic modelling technique using Neural Topic Model, Variational Autoencoders can be applied to study its impact on the model. Also, the application of this model in real time data with some modification in the work to feed real time data will be great.

References:

[1] Velickovic P., Cucurull G., Casanova Ar., Romero Ad., Liò P. & Bengio Y. 2018. “Graph Attention Networks”. ICLR (2018).

[2] X. Wang, Ji H. Y., Shi C., Wang B., Cui Pe., Yu P. & Yanfang Ye. 2019. “Heterogeneous graph attention network”. In WWW.

[3] H. Linmei, T. Yang, C. Shi, H. Ji, and X. Li. 2019. “Heterogeneous graph attention networks for semi-supervised short text classification”. EMNLP,

[4] Cao M., Ma X., Xu M., Wang C. 2019. “Heterogeneous Information Network Embedding with Meta-path Based Graph Attention Networks”. ICANN, Lecture Notes in Computer Science, vol 11731. Springer, Cham.

[5] Vaswani As., Shazeer N., Parmar Ni., Uszkoreit J., Jones L., Gomez Ad., Kaiser Lu. & Polosukhin I. 2017. “Attention is All you Need”. In NIPS. 5998–6008.

[6] Zhang Xi., Zhao Ju. & LeCun Ya. 2015. “Character-level convolutional networks for text classification”. In NIPS. 649–657

[7] Mikolov To., Sutskever Il., Chen K., Corrado G. & Dean Je. 2013. “Distributed Representations of Words and Phrases and their Compositionality”. In NIPS.

[8] Kiran K. T., Wang Ch., Oh Se. & Li Li. 2018. “Attention-based Graph Neural Network for Semi-supervised Learning”

[9] Wu Zo., Pan Sh., Chen Fe., Long Gu., Zhang Ch. & Yu Ph. S. 2019. “A Comprehensive Survey on Graph Neural Networks”. IEEE

[10] Zhou Ji., Cui Ga., Zhang Z., Yang Ch., Liu Zh., Wang Li., Li Ch. & Sun Ma. 2019. “Graph Neural Networks: A Review of Methods and Applications”

[11] Yao Li., Mao Ch. & Luo Yu. 2019. “Graph convolutional networks for text classification”. AAAI.

[12] Kipf Th. N. & Welling M. 2017. “Semisupervised classification with graph convolutional networks”. In ICLR.

[13] Rousseau Fr., Kiagias Em. & Vazirgiannis Mi. 2015. “Text categorization as a graph classification problem”. In ACL, volume 1, 1702–1712.

[14] Wang J., Wang Zh., Zhang Da. & Yan Ju. 2017. “Combining knowledge with deep convolutional neural networks for short text classification”. In IJCAI, volume 350.

[15] Blei D.M., Ng A.Y. & Jordan M. I. 2003. “Latent Dirichlet allocation”. Journal of machine Learning research, 3(Jan): 993-1022.

[16] Kalchbrenner N., Grefenstette E. & Blunsom P. 2014. “A convolutional neural network for modelling sentences,”, ACL 2014 - Proceedings of the Conference.

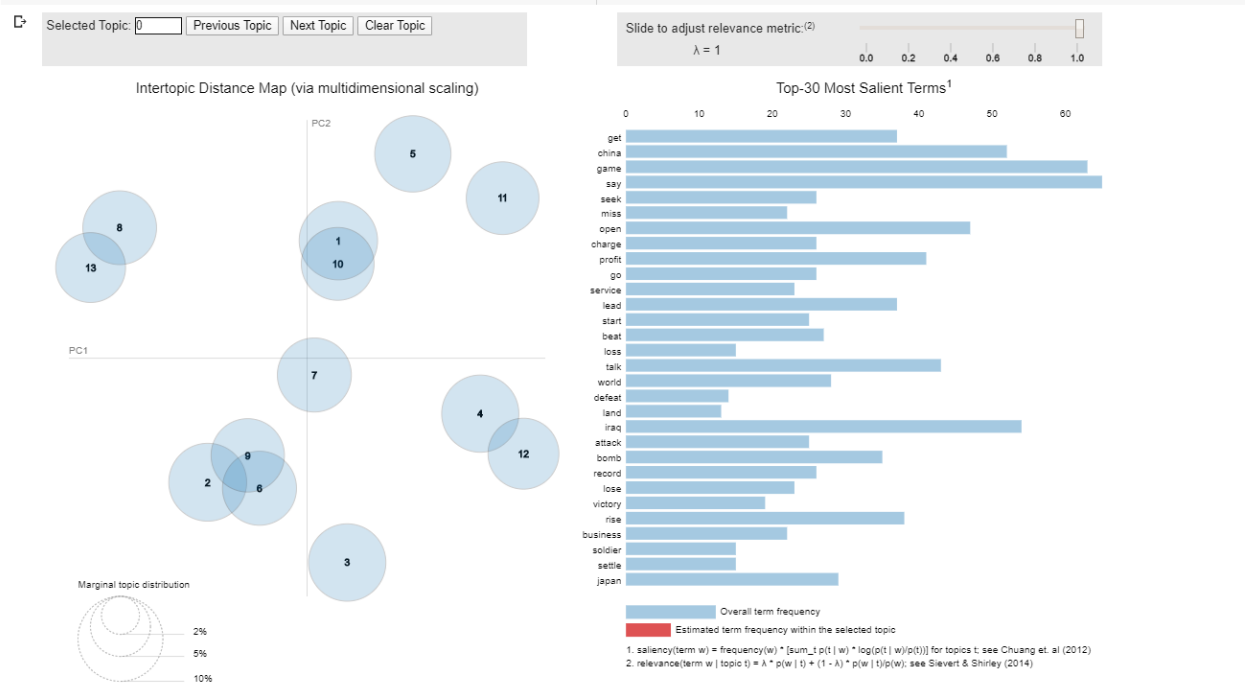
[17] Kim Y. 2014. “Convolutional neural networks for sentence classification,” in EMNLP 2014, Proceedings of the Conference.

[18] Brokos G., Malakasiotis P. & Androutsopoulos I. 2016. “Using Centroids of Word Embeddings and Word Mover’s Distance for Biomedical Document Retrieval in Question Answering”. 15th Workshop on Biomedical NLP. pg: 114–118

[19] [Online]. Available: https://www.kaggle.com/rmisra/news-category-dataset
[20] [Online]. Available: https://github.com/mhjabreel/CharCnn_Keras/tree/master/data

APPENDIX A

Each topic visualization with PyLDAVis



APPENDIX B

Dominant topic in each document

```
df_dominant_topic.head(10)
```

	Document_No	Dominant_Topic	Topic_Perc_Contrib	Keywords	Text
0	0	5.0	0.8707	price, microsoft, warn, sell, say, talk, bankr...	Derrida, father of deconstruction, dies at 74
1	1	15.0	0.9303	home, seek, hit, help, oracle, battle, peoples...	UN hostages held for 4 weeks in Afghanistan freed
2	2	6.0	0.4949	drug, test, poll, rise, see, settle, sales, go...	House GOP Seeks to Quash Draft Rumor
3	3	10.0	0.9044	win, beat, miss, soldier, moon, rebel, earn, e...	Telstra, Not Government, To Choose New Chief
4	4	12.0	0.6731	face, lose, step, apple, find, sony, tech, ira...	Cricket crisis: Reform or face the music
5	5	14.0	0.9367	open, give, victory, sue, approve, business, e...	Kuznetsova struggles to advance to quarters in...
6	6	9.0	0.5456	iraq, bomb, group, season, make, offer, hole, ...	Car bombs claim a dozen lives in Iraq
7	7	10.0	0.9304	win, beat, miss, soldier, moon, rebel, earn, e...	Krill decline raises concern for Antarctics fo...
8	8	15.0	0.9302	home, seek, hit, help, oracle, battle, peoples...	J.C. Penney Posts Second-Quarter Profit
9	9	6.0	0.8707	drug, test, poll, rise, see, settle, sales, go...	SMART-1 settles into lunar orbit

APPENDIX C

Top document and Topic distribution over documents

Topic No.	Topic_Perc_Contrib	Keywords	Text	Num_Documents	Document Percentage
0.0	0.9462	kill, china, vote, update, bank, charge, reute...	Indian software major Wipro second-quarter net...	908	15.13
1.0	0.9422	report, get, reuters, study, mobile, strike, t...	World ; Death Toll from Israel #39;s Gaza Offe...	368	6.13
2.0	0.9461	time, reuters, india, yankees, hold, security,...	Strike talk heats up as hearing on US Airways ...	397	6.62
3.0	0.9422	deal, sign, week, attack, bush, ready, microso...	New chip designs help AMD boost market share	340	5.67
4.0	0.9460	profit, look, fight, wireless, need, post, say...	Barclays says on track to meet full-year profi...	352	5.87
5.0	0.9423	price, microsoft, warn, sell, say, talk, bankr...	Google Sells Shares in IPO for \$85 After Cutt...	309	5.15
6.0	0.9423	drug, test, poll, rise, see, settle, sales, go...	SPORTS OF THE TIMES Clemens Meets Match at Han...	333	5.55
7.0	0.9495	take, plan, china, buy, say, good, clash, game...	China tries to shut down phone sex lines in an...	304	5.07
8.0	0.9459	game, claim, file, rule, record, japan, flight...	India clears cricket team to begin Bangladesh ...	353	5.88
9.0	0.9424	iraq, bomb, group, season, make, offer, hole, ...	Oracle makes \$9.2 billion take-it-or-leave-it...	293	4.88
10.0	0.9462	win, beat, miss, soldier, moon, rebel, earn, e...	Norway's Shipowners plans to expand lockout, g...	356	5.93
11.0	0.9425	play, launch, go, world, year, music, linux, n...	British Troops Head North for Mission Near Bag...	329	5.48
12.0	0.9424	face, lose, step, apple, find, sony, tech, ira...	Iran Tells Russia to Expand Nuclear Ties	350	5.83
13.0	0.9423	talk, lead, peace, state, hop, work, google, h...	Health, financial firms top Working Mother #39...	352	5.87
14.0	0.9424	open, give, victory, sue, approve, business, e...	Business ; India #39;s growth target lowered, ...	325	5.42
15.0	0.9463	home, seek, hit, help, oracle, battle, peoples...	Windows may be free in Iran, but security fear...	331	5.52

APPENDIX D

Test on Tagmynews Dataset

Train per class	Train data	Validation data	Test data	Accuracy	Epoch
40	200	200	5600	43.78	93
80	400	400	5200	51.30	115
120	600	600	4800	56.5	62
160	800	800	4400	59.57	85
200	1000	1000	4000	60.1	96

APPENDIX E

Topic test in Tagmynews and Agnews dataset α, β and coherence score

Topics	Alpha	Beta	Coherence (Tagmynews)	Coherence (Agnews)
8	0.01	0.01	0.423283188	0.668765447
8	0.01	0.91	0.432987171	0.661578514
8	0.01	0.125	0.427991433	0.663935707
8	0.91	0.01	0.246284092	0.601170789
8	0.91	0.91	0.239600747	0.616772982
8	0.91	0.125	0.246284092	0.604931634
8	0.125	0.01	0.429949947	0.676541471
8	0.125	0.91	0.435737779	0.680123849
8	0.125	0.125	0.426510959	0.678046934
9	0.01	0.01	0.448243868	0.681141614
9	0.01	0.91	0.425878167	0.667646666
9	0.01	0.111111	0.439021298	0.671927332
9	0.91	0.01	0.239399251	0.613368798
9	0.91	0.91	0.23565994	0.614484321
9	0.91	0.111111	0.252563766	0.607765119
9	0.111111	0.01	0.457078765	0.67034315
9	0.111111	0.91	0.449215432	0.666086717
9	0.111111	0.111111	0.449228691	0.668708829
10	0.01	0.01	0.446557611	0.676760279
10	0.01	0.91	0.457060118	0.672900664
10	0.01	0.1	0.443131957	0.676922558
10	0.91	0.01	0.23203547	0.608771854
10	0.91	0.91	0.225666867	0.614428949
10	0.91	0.1	0.23203547	0.612564136
10	0.1	0.01	0.462082403	0.684399057
10	0.1	0.91	0.431106714	0.6821025
10	0.1	0.1	0.455574618	0.682106633
11	0.01	0.01	0.476666057	0.697755831
11	0.01	0.91	0.453828935	0.694268412
11	0.01	0.090909	0.47030032	0.697667252
11	0.91	0.01	0.342765872	0.606235778
11	0.91	0.91	0.225033396	0.611440437
11	0.91	0.090909	0.228094328	0.606235778
11	0.090909	0.01	0.485218992	0.701536874
11	0.090909	0.91	0.470583796	0.695038754
11	0.090909	0.090909	0.47475655	0.699588901

12	0.01	0.01	0.489698858	0.697683636
12	0.01	0.91	0.491226479	0.688488065
12	0.01	0.083333	0.491569199	0.695550308
12	0.91	0.01	0.23224239	0.601661
12	0.91	0.91	0.229504318	0.611515804
12	0.91	0.083333	0.23224239	0.601660822
12	0.083333	0.01	0.486686944	0.697909438
12	0.083333	0.91	0.484376412	0.69736253
12	0.083333	0.083333	0.489298264	0.697764298
13	0.01	0.01	0.491509155	0.689127588
13	0.01	0.91	0.486032504	0.685708962
13	0.01	0.076923	0.492631387	0.685345373
13	0.91	0.01	0.239873685	0.603288644
13	0.91	0.91	0.23406042	0.599048667
13	0.91	0.076923	0.234506656	0.603288644
13	0.076923	0.01	0.498184532	0.697007748
13	0.076923	0.91	0.484367185	0.694875702
13	0.076923	0.076923	0.49857289	0.698205143
14	0.01	0.01	0.482690732	0.710691217
14	0.01	0.91	0.480095556	0.703665704
14	0.01	0.071429	0.480181929	0.709923901
14	0.91	0.01	0.235091085	0.599391532
14	0.91	0.91	0.229277366	0.607766275
14	0.91	0.071429	0.282414551	0.599736328
14	0.071429	0.01	0.496306676	0.710380061
14	0.071429	0.91	0.483590531	0.706648082
14	0.071429	0.071429	0.492255321	0.70843945
15	0.01	0.01	0.497303752	0.700808844
15	0.01	0.91	0.502133282	0.691371349
15	0.01	0.066667	0.488546948	0.698464318
15	0.91	0.01	0.231454663	0.602409438
15	0.91	0.91	0.226169949	0.61023896
15	0.91	0.066667	0.23255767	0.602601446
15	0.066667	0.01	0.519161757	0.699975179
15	0.066667	0.91	0.496582198	0.69261238
15	0.066667	0.066667	0.516043101	0.701114875
16	0.01	0.01	0.503876029	0.698658332
16	0.01	0.91	0.499837691	0.696425547
16	0.01	0.0625	0.504434346	0.699901743
16	0.91	0.01	0.390620621	0.595199577
16	0.91	0.91	0.224559204	0.615180937
16	0.91	0.0625	0.231746928	0.601943621
16	0.0625	0.01	0.501416444	0.697708496

16	0.0625	0.91	0.500615194	0.687028555
16	0.0625	0.0625	0.496223254	0.698229081
17	0.01	0.01	0.520535653	0.701847711
17	0.01	0.91	0.513736164	0.694217987
17	0.01	0.058824	0.521653141	0.698841097
17	0.91	0.01	0.232203721	0.60151882
17	0.91	0.91	0.218717353	0.604531929
17	0.91	0.058824	0.228343292	0.597811803
17	0.058824	0.01	0.523768793	0.697781218
17	0.058824	0.91	0.510507106	0.690338071
17	0.058824	0.058824	0.520626773	0.696051391
18	0.01	0.01	0.504067337	0.691231525
18	0.01	0.91	0.495153999	0.688852313
18	0.01	0.055556	0.49651153	0.692429387
18	0.91	0.01	0.233498247	0.600307215
18	0.91	0.91	0.223866107	0.605981407
18	0.91	0.055556	0.231370854	0.601171131
18	0.055556	0.01	0.5048067	0.693414442
18	0.055556	0.91	0.498500856	0.686705297
18	0.055556	0.055556	0.506236873	0.696116861
19	0.01	0.01	0.519632118	0.699141979
19	0.01	0.91	0.518462781	0.6913261
19	0.01	0.052632	0.523370241	0.699611078
19	0.91	0.01	0.227437021	0.598781224
19	0.91	0.91	0.220421457	0.611749247
19	0.91	0.052632	0.229134802	0.599342439
19	0.052632	0.01	0.5221799	0.693144714
19	0.052632	0.91	0.506389825	0.69821828
19	0.052632	0.052632	0.517393237	0.692279713
20	0.01	0.01	0.516235758	0.695628344
20	0.01	0.91	0.507668487	0.67939375
20	0.01	0.05	0.525942563	0.692847069
20	0.91	0.01	0.234435978	0.597155561
20	0.91	0.91	0.221807081	0.611657425
20	0.91	0.05	0.231721052	0.597022713
20	0.05	0.01	0.515073507	0.703455979
20	0.05	0.91	0.519116465	0.690424391
20	0.05	0.05	0.509254868	0.699282836

Updated Final Thesis Report/Updated Final Thesis Report/074MSCSK_013_SUJIL.pdf

ORIGINALITY REPORT

16%
SIMILARITY INDEX

PRIMARY SOURCES

1	flipkarma.com Internet	234 words — 3%
2	aclanthology.org Internet	115 words — 1%
3	Tianchi Yang, Linmei Hu, Chuan Shi, Houye Ji, Xiaoli Li, Liqiang Nie. "HGAT: Heterogeneous Graph Attention Networks for Semi-supervised Short Text Classification", ACM Transactions on Information Systems, 2021 Crossref	56 words — 1%
4	pdfs.semanticscholar.org Internet	53 words — 1%
5	Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, Philip S Yu. "Heterogeneous Graph Attention Network", The World Wide Web Conference on - WWW '19, 2019 Crossref	42 words — < 1%
6	library.oapen.org Internet	42 words — < 1%
7	citeseerx.ist.psu.edu Internet	37 words — < 1%