DYNAMIC SPATIO-TEMPORAL GRAPH NEURAL NETWORKS FOR HOT TOPIC PREDICTION IN SCIENTIFIC LITERATURE

A Thesis

Presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

YIJIE REN

Dr. Dong Xu, Thesis Supervisor

MAY 2020

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

DYNAMIC SPATIO-TEMPORAL GRAPH NEURAL NETWORKS FOR HOT TOPIC PREDICTION IN SCIENTIFIC LITERATURE

presented by Yijie Ren,

a candidate for the degree of master of science,

and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Dong Xu

Dr. Trupti Joshi

Dr. Fang Wang

ACKNOWLEDGEMENTS

I would first like to thank my thesis supervisor, Professor Dong Xu, for all of his professional guides and persistent supports throughout my master study. Once my mentor, always my mentor. His passion and professional attitude towards academic research will continue to inspire me through my future academic life. I would also like to extend my sincere appreciation to Dr. Xiaohu Shi, a visiting scholar in Digital Biology Laboratory (DBL) from Jilin University, China, for his suggestions and guidance on my thesis. He was always there to help me when I ran into a dead end on my research and steered me in the correct direction. The accomplishment of my research would not have been possible without his help. At last, I want to express my gratitude to Shuai Zeng, a PhD student in DBL, for providing me with detailed technical explanations and continuous encouragements.

TABLE OF CONTENTS

LIST OF FIGUERS

LIST OF TABLES

ABSTRACT

With information explosion occurring in past decades, the rapid growth of papers published results in the rapid change of hot topics, especially in the biomedical domain. It turns out very hard for researchers who are interested in biomedical domain to track hot topics over time, as well as to predict the trends of them in the near future. Based on the above demand, it is important to have a model which is able to follow and predict the trend of hot topics continuously. Deep learning has been proven to be an efficient method to extract information from texts and use the information to predict the future trends. Under the thriving background of Deep Learning, Graph Neural Network (GNN) is able to capture the information from graph structures. There are various applications using GNN models, such as traffic flow prediction, chemical structure discovering, etc. In this research project, a dynamic spatio-temporal graph neural network is presented to keep track of the selected hot keywords and topics in the biomedical domain and predict the possible frequencies in the near future. The input of the model is obtained by extracting the monthly frequency information of selected keywords and topics from paper abstracts in PubMed, the largest biomedical literature collection. After training with data over a decade, the model is able to predict trends of selected hot keywords and topics in next 5 months. Thus, the presented model can help follow the trend of hot topics in the biomedical domain.

Chapter 1 Introduction

1.1 Background

As human beings entered Information Age, the rapid growth of computing technology induced the generation of manifold data, as well as the ability to process data with high speed, both resulting in information explosion in all domains. Obviously, biomedical domain cannot be an exception. In fact, biomedical structures, such as genes and proteins, carry huge biomedical information inside the structures of themselves, and that is exactly the reason why study of biomedical information raised researchers' interests first.

With the emergence of numerous kinds of models entailed by the development of computing technology, the research passion has been rising fast among biomedical researchers in different subdomains of biomedicine. Thus, papers related to biomedical researches have been growing dramatically in past decades. Due to the tremendous amounts of papers in biomedical domain, the research topics inside biomedical domain may shift quickly and sometimes are not easy to follow. Fortunately, this change can be reflected by abstracts of published papers in time-series. It is vital to keep tracking trends of hot research topics for every biomedical researcher during their research careers. Also, it can help new biomedical researchers or people who are simply interested in this area take a general look at possible variation of hot research points in the near future within biomedical domain, which might give them suggestions of the topic(s) they would like to follow from now on. It is always a good way to take a look at how hot keywords and topics vary from time to time in biomedical domain, and learn about the possible future

trend, before one starts to do research or project related to a certain topic.

1.1.1 Brief Introduction of Data Preprocessing

With papers published in recent years, it is possible for us to find out the research topic variation and give predictions appropriately. In this project, we retrieved published biomedical papers from PubMed [1], the largest collection of biomedical literature, and preprocessed the papers by extracting the information from each paper of title, keyword, abstract and publication date. Hot keywords and hot topics are determined from the extracted paper information, along with the monthly frequency of each hot keyword/topic as temporal information. The trends of hot keywords and topics can be reflected by the times of occurrence in the next several months. To utilize the spatial information of hot keywords or topics, a graph consisting of the co-occurrence information is constructed among these keywords and topics in each month. Eventually, there are 3 datasets of hot keywords and topics, each consisting of corresponding frequencies and co-occurrence matrices. The details of data preprocessing and dataset construction are available in Chapter 2.

1.1.2 Brief Introduction of Model and Result

Deep Learning is a subdomain of Machine Learning, and it has raised a lot of attentions due to the explosive developments in recent years. Inspired by human brains, Deep Learning simulates the structures and functions as large neural networks with numbers of layers and multiple different activation functions inside each hidden neuron.

There are various types of Deep Learning structures which can be fit into various tasks, such as supervised learning [2], unsupervised learning [3], reinforcement learning [4], etc. Also, there exist multiple kinds of activation functions, such as sigmoid [5] and ReLU [6]. With training of large datasets, it has been proven that Deep Learning is an efficient method and has achieved a lot of success in plenty of different applications within different domains, such as speech recognition [7], natural language processing[8], etc.

Among all the Deep Learning applications, Convolutional Neural Networks (CNN) can be considered as a very popular architecture. There are numerous applications utilizing CNN architectures, which are served for different purposes, such as human action recognition [9], image recognition [10], etc. Apart from the above applications, CNN is also able to collect temporal information, which is called causal convolution (the detailed explanation can be found in Section 3.1.2). The causal convolution in this paper is able to acquire temporal information for every hot keyword and topic. After obtaining the temporal information, the spatial information can be obtained with the help of Graph Neural Networks.

Based on the boom of Deep Learning, the research related to Graph Neural Networks, a branch of Deep Learning, has been growing fast as well. Unlike traditional data inside Euclidean domain, GNN focuses on acquiring information from data inside non-Euclidean domain, i.e. graph structure. With the pattern extraction from self-node and neighbor nodes within several hops, a specially designed graph convolution can integrate information from a node itself and neighbors of that certain node, which is called Graph Convolution Network (GCN). With the characteristics of GCN, we can

obtain the spatial information of hot keywords/topics. After integrating the spatial information into the temporal information, we are able to predict the possible trends of hot keywords and topics. It is also a novel trial to use GNN as part of a model to predict hot topic trends in biomedical domain. The detailed explanations of GCN and the model we used are available in Chapter 3.

With historical frequencies of hot keywords and topics in past 15 months as the input of model, the experiments are conducted based on 3 datasets. The results of predicted frequencies in next 5 months of hot keywords and topics are demonstrated based on Pearson correlation coefficient and the hit accuracy in top fluctuated keywords/topics. The predictions are compared with direct copy of corresponding keyword/topic frequencies in last one month from historical 15 months, which is considered as the benchmark of result evaluation. After comparison, the average results of predictions on each dataset outperforms the defined benchmark. Besides the results of all hot keywords/topics, some predictions of specific keywords/topics are analyzed in a continuous time series as well. All of the results and associated analysis are available in Chapter 4.

1.2 Literature Review

Under the prosperity of Deep Learning, Graph Neural Networks have been utilized in various disciplinary researches, such as chemistry [11], biology [12], etc. A comprehensive survey conducted by IEEE Fellow [13] classifies GNN into 5 different categories based on the neural network structures – Graph Convolution Networks, Graph Attention Networks [14], Graph Auto-encoders [15], Graph Generative Networks [16]

and Graph Spatial-temporal Networks [17]. The last 4 categories are all intersected with Graph Convolution Networks, which indicates GCN is the core of GNN. From Background section above, it can be inferred that this research project takes advantages of both temporal and spatial information, therefore, a Graph Spatial-temporal Network is adopted to achieve the goal.

Graph Spatial-temporal Networks are utilized in many applications, such as traffic forecasting, action recognition, etc. For example, Shi et al. [18] proposed a directed graph neural network to recognize human actions based on the skeleton data of human body formed as directed acyclic graph (DAG), while utilizing CNN to capture the temporal information. Yan et al. [19] also presented a model to recognize human action, called Spatial-Temporal Graph Convolutional Networks (ST-GCN), based on dynamic skeletons. Li et al. [20] presented a Diffusion Convolutional Recurrent Neural Network (DCRNN) to forecast traffic flow with Recurrent Neural Networks (RNN) to collect traffic temporal information. Yu et al. [21] also proposed a model for traffic forecasting purpose, named Spatio-Temporal Graph Convolutional Network (STGCN), but leveraging CNN to acquire traffic temporal information. Based on the above research works, it can be confirmed that GCN is always leveraged to collect spatial information, accompanying with CNN/RNN to collect temporal information in Graph Spatial-temporal Networks.

Apart from GCN, trend prediction analysis is also a focus which needs to be addressed. Trend prediction analysis has been prevailing in a long time, as well as in many domains. First, there is plenty of research trying to predict stock trends. Naeini et al. proposed models using Multi Layer Perceptron and Elman recurrent neural network to

predict stock market [22]. Fung et al. predicted stock market by proposing a model combining event-knowledge and neural networks [23]. Hong and Han proposed a neural network model with cognitive maps to predict stock [24]. Other than stock prediction, there is also research related to crime trend prediction, such as predictions of crime hotspot. Many different techniques are applied to predict crime trend. For example, Kianmehr and Alhajj proposed a model employed Support Vector Machine (SVM) [25], while Liao et al. utilized Bayesian network [26]. Also, Corcoran et al. presented a neural network model to predict geographical crime [27]. In medical domain, virus trends can be analyzed based on statistical methods [28]. Besides the above categories of trend predictions, there are also trend predictions of cyber security incidents, with the help of Machine Learning methods [29]. Similar to this paper, topic trend prediction becomes more important with the prevalence of social media, especially microblogging system like Twitter. Based on text contents inside social media, public opinion analysis can be conducted with the help of topic trend prediction. There are many research papers related to public opinion analysis. Chen et al. proposed a model based on contents and social connections [30], while Wang et al. proposed a model based on Grey Verhulst Model [31]. Also, a collaborative filtering based topic trend prediction model is presented by Chen et al. [32].

1.3 Goal of the Study

Due to the copious keywords of PubMed Literature, it is quite difficult to come up with a method to determine hot keywords/topics. This is the first addressed problem in this project. After comparing several different methods, mean and standard deviation are

adopted to filter out the plain keywords while keeping the hot ones, i.e. select keywords satisfying the higher values of certain mean and standard deviation. Then further selection steps based on standard deviations from window slicing through time series are applied to obtain final datasets. Meanwhile, the hot topics are clustered from selected hot keywords based on proper topic clustering model. Appropriate data preprocessing method is able to provide with appropriate input data, but in order to predict the trends of them, the model is rather more important.

Since both temporal and spatial information of hot keywords/topics are utilized in this project, a model is needed to complete the above task. After investigating the state-of-art Graph Spatial-temporal Network models, the last model mentioned in Literature Review section named Spatio-Temporal Graph Neural Networks [21] is selected as the final model. The original STGCN model is used to predict traffic flow, which collects spatial information from road routes as a static graph. However, the graphs used in this project are various from one month to another, which means the spatial information is dynamic. Therefore, instead of utilizing the exact same model, we modified the static GCN in this model into a dynamic one, in order to address this problem better, which can also achieve the goal of this research project.

The hot keywords and topics are selected based on appropriate methods. After finalizing the appropriate input datasets, the proper model is also selected based on ability of integrating temporal and spatial information of each dataset. Among all hot keywords and topics, temporal information can be collected easily by counting occurrence of each hot keyword/topic in each month, while the spatial information can be underlined by co-occurrence of these keywords/topics in each month. The model which takes full

advantage of these 2 kinds of information is adopted from state-of-art models as well. After addressing the above 2 main problems, with the input from large literature collection in PubMed, the modified model is able to predict trends of hot keywords/topics in the next few months.

Chapter 2 Data Preprocessing

2.1 Data Source

PubMed [1] is a website consisting of the largest literature of biomedical and life sciences. Because of such abundance of biomedical data, this project eventually adopted to collect biomedical papers from PubMed. Papers can be retrieved from the access link (ftp://ftp.ncbi.nlm.nih.gov/pub/pmc/oa_bulk/) provided by PubMed. The papers used in this project are published from January 1997 to December 2016, 240 months in total, as the whole dataset, while there are some datasets only utilize part of the whole dataset. There are 989,067 papers in total. Inside each downloaded paper, 4 sections were extracted – title, keyword, abstract and publication date. After obtaining the above 4 sections, title, keyword and abstract are integrated as the new paper content for each paper.

2.2 Data Formation and Preprocessing

After collecting all of the keywords in each paper's keyword section, the final keyword set contains 549,619 keywords in total, as the whole vocabulary list. It is important to keep in mind that not all keywords are consisting of single word, most of them are phrases, such as clinical cancer research, protein structure, etc. In the final prediction, normalized word frequencies for each keyword/topic of the next 5 months are the actual predictions which can reflect the trends of hot keywords or topics. Word frequency is acquired monthly for all 549,619 keywords by counting how many times a certain keyword appears in all paper contents of that month. Thus, every keyword

contains 240 frequency points.

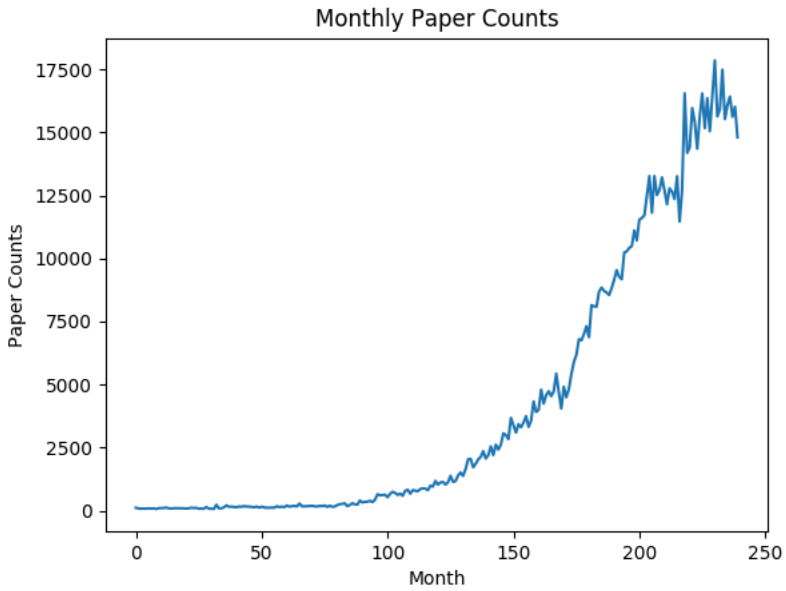


Figure 1Published paper amounts in each month from May 2005 to December 2016, 140 months in total.

Among all the keywords, those with low occurrence and/or low fluctuation are not considered as the hot keywords, since it is meaningless to predict the trends of keywords which appear only once or twice in the whole dataset. For each keyword, the frequency in each month is increasing due to the increasement of published papers. The increasement of published papers in each month is shown clearly in Figure 1. Because of the great difference of paper amounts from month to month, keyword frequencies grow dramatically as well. It turns out that simple keyword frequency may not be competent to be the feature of each keyword, instead, the proportion of each keyword frequency over all keyword frequencies in each month is a better way to represent the corresponding "frequency". The newly-defined "frequency" is named as keyword frequency proportion.

The keyword frequency proportion of keyword index j, $P_{i,j}$ in a certain month i, $i \in \{1,2,\dots,240\}$, $j \in \{1,2,\dots,549619\}$, can be defined as,

$$P_{i,j} = \frac{o_{i,j}}{\sum_j o_{i,j}} \quad (1)$$

where $O_{i,j}$ is the frequency of keyword index j in month i. After calculating the mean and standard deviation of each keyword frequency proportion throughout the 240 months, keywords with lower mean and/or lower standard deviation were removed. As a result, the final size of vocabulary list becomes 301,363.

The purpose of this project is to predict trends of hot keywords/topics in biomedical domain, where hot keywords or topics are defined as keywords/topics with high fluctuation through the time series. The keyword frequencies of whole 240 months can be represented as a matrix of 240 months * 301,363 keywords, with row as month, column as keywords, see Figure 2 for illustration. In order to obtain hot keywords/topics from final vocabulary list, it is necessary to split the dataset represented as 240 months * 301,363 keywords into 211 different keyword frequency chunks by applying a 30-months-high window slicing through the whole time series, shown in Figure 2. There are T rows (240 months) in total and N columns (301,363 keywords) in total. For example, $O_{3,2}$ means the keyword frequency of 2nd word in 3rd month. After window slicing, there are 211 chunks of datasets, with each chunk of dataset as a 30 months * 301,363 keyword frequency matrix.

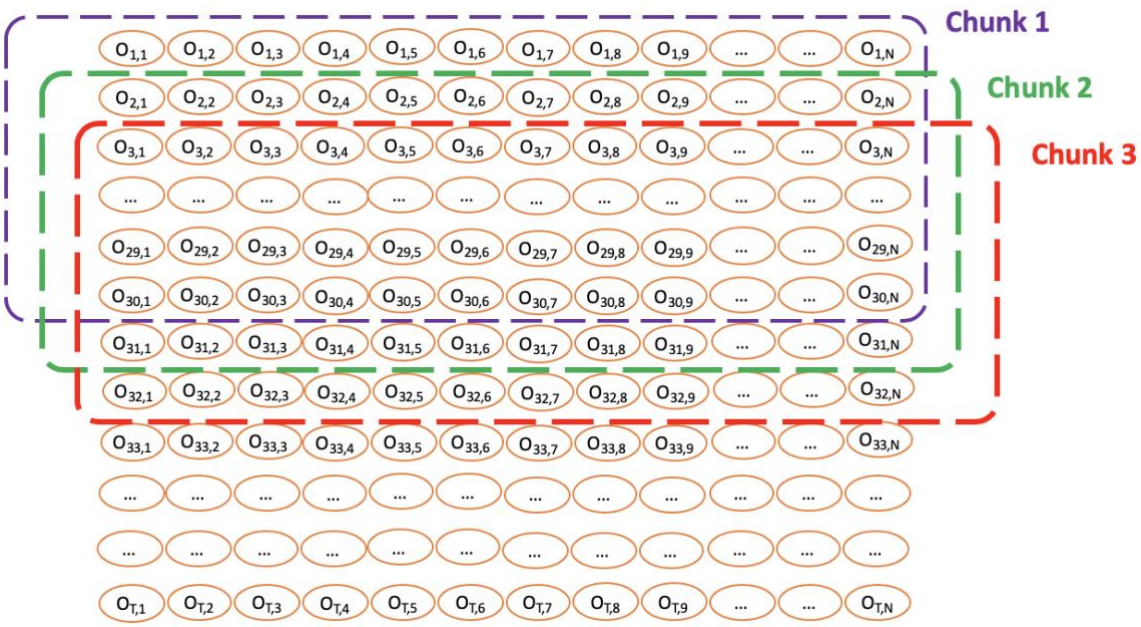


*Figure 2 Window slicing through whole 240 months keyword occurrences. There are T rows (240 months) and N columns (301,363 keywords) in total. After window slicing, there will be 211 chunks of datasets, each as a 30*301,363 matrix.*

Inside each chunk of dataset, standard deviations of each keyword are calculated based on each keyword frequency proportion within 30 continuing months, then there are 211 standard deviations for each keyword in total. Among 211 standard deviations of each keyword, the largest one is selected to be the representative standard deviation for each keyword, with the descending order applied afterwards. After selecting the keywords with highest 116 and highest 642 standard deviations, 2 different keyword datasets are formed by these keywords. Due to the small paper amounts in previous 100 months (only several hundred each month), only papers in last 140 months (from May 2005 to December 2016) are adopted to be the final data source. Thus, the keyword frequency proportions of the last 140 months are considered to be included in 2 keyword datasets. The 2 datasets are labeled as **Keywords(S)** and **Keywords(L)**, according to the

small/large size of each dataset. There is another **Topic** dataset used in the project, which is introduced below in section 2.2.2.

2.2.1 Keyword Dataset Formation and Preprocessing

After finalizing hot keywords and associated keyword frequency proportions, it is important to form the temporal and spatial information of these hot keywords in each month and append the information to both Keywords(S) and Keywords(L). In each dataset, keyword frequency proportions of 15 continuing months are used as historical data to predict frequency proportions in next 5 months separately. To be more specific, there are keyword frequency proportions of 20 continuing months in every input example, with first 15 months as historical data, last 5 months as ground truth of predictions. To satisfy the requirement of 20 continuing months, a 20-months-high window is slicing through the whole dataset (140 months), similar to Figure 2, and 121 examples are acquired after window slicing. In both Keywords(S) and Keywords(L), the training, validation and testing datasets are formed by randomly selecting the examples from 121 examples, with no overlapping. Historical data (15 months) in every example is normalized by Z-score method.

Unlike the temporal features of each hot keyword, spatial information relies on the graph constructed by these selected keywords. Each hot keyword can be regarded as a node in graph, with keyword frequency proportion as its sole feature. Since the standard deviations of all selected keywords are pretty high, it results in isolation of nodes in the graph, see left part of Figure 3. In order to better extract the spatial information from these standalone nodes, some add-on nodes are integrated into the graph, in order to

contribute more connections among these hot keywords, i.e. add more edges into the graph, see Figure 3. Therefore, 1500 other add-on keywords from final vocabulary list (selection criteria are introduced in next paragraph) have been chosen to enrich the spatial information of selected 116/642 hot keywords. Different datasets have different selected add-on keywords, so each dataset is accompanied with other different 1500 add-on keywords to form the graph. However, the trends of these add-on keywords are not expected to be predicted, while they only provide with the add-on spatial information to hot keywords.

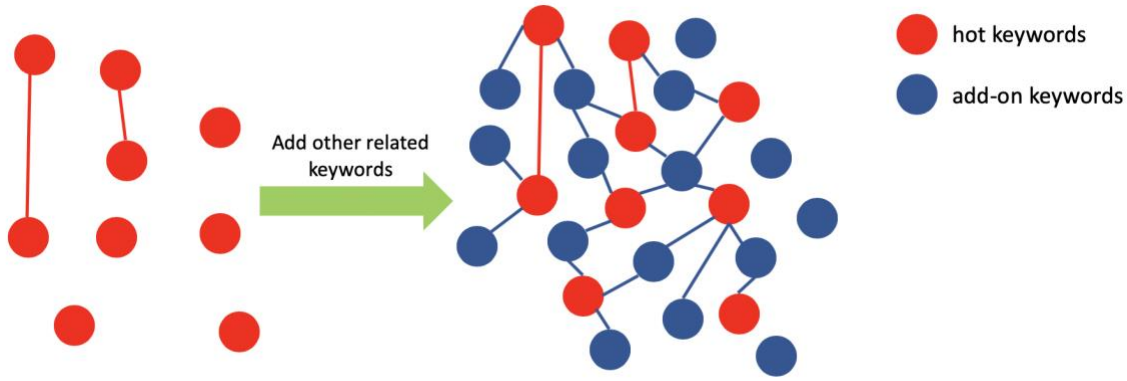


Figure 3 Graph structure change after adding 1500 other keywords.

In order to select 1500 add-on keywords with the greatest contributions to 116/642 hot keywords, graph edge connections with hot keywords inside the whole 140 months are adopted as the criteria, i.e. the degree of each add-on keywords with 116/642 hot keywords. After obtaining all 1500 add-on nodes with highest degrees, the weights of edge between two nodes are defined as the co-occurrence times in all the papers within a certain month. Eventually, different graphs of 1616/2142 nodes (keywords) can be

acquired in different months.

2.2.2 Topic Dataset Formation and Preprocessing

The Topic dataset is formed by Keywords(L) with 642 keywords. The reason why only Keywords(L) is chosen is that Keywords(S) contains only 116 keywords, which is too small to be clustered as multiple topics.

Latent Dirichlet Allocation (LDA) model [33] is applied to form 25 topics from 642 keywords. LDA model is well known for its topic modelling over a set of documents, while it can also be applied to a set of words. As for the procedure, LDA supposes there are a fixed amount of topic(s), and it assigns words to each topic during the model training process. After completing the training process, every topic is represented by a list of all input words, along with a probability distribution over all words, where the probability distribution indicates the probability that a certain word is assigned to a certain topic.

When it comes to Keywords(L) dataset, there is a probability distribution over 642 keywords along with each topic after applying LDA model. By selecting the keywords with top 25 probabilities inside each topic, a total vocabulary list of 524 keywords can be generated by removing duplicated keywords in 625 keywords obtained from 25 topics * 25 top probability keywords. Thus, for each topic, there is a probability list of 524 keywords. After collecting the keyword frequencies of each keyword inside 524 keywords for each month, each keyword frequency is multiplied by the corresponding probability, then a vector with length of 524 is aligned to each topic as topic vector for each month.

Similar to calculation of keyword frequency proportion in Keywords(S) and Keywords(L), we define topic frequency T_i, $i \in \{1,2,\dots,25\}$, in a certain month as summation of the topic vector in that month, calculated as,

$$T_i = \sum_j F_{i,j} P_{i,j}\,, j \in \{1,2,\dots,524\} \quad (2)$$

where $F_{i,j}$ and $P_{i,j}$ are the keyword frequencies and corresponding probabilities under a certain topic i in each month, respectively. Therefore, considering 25 topics as 25 nodes in the graph, the topic frequency defined above is treated as the sole feature of each topic in each month.

As for graph construction for Topic dataset, the 25 nodes are considered fully connected with each other, i.e. all 15 nodes are formed as a connected graph. The weight of each edge between 2 nodes (indexed as i and j) is defined as the Euclidean distance of 2 topic vectors, written as,

$$W_{i,j} = \sqrt{\sum_k (V_{i,k} - V_{j,k})^2} \quad (3)$$

where k is the element index in each topic vector, $k \in \{1,2,\dots,524\}$.

2.3 Data Preprocessing Architecture

From above introduction of 3 datasets, an architecture diagram is attached below to illustrate the process in a more straightforward way, see Figure 4.

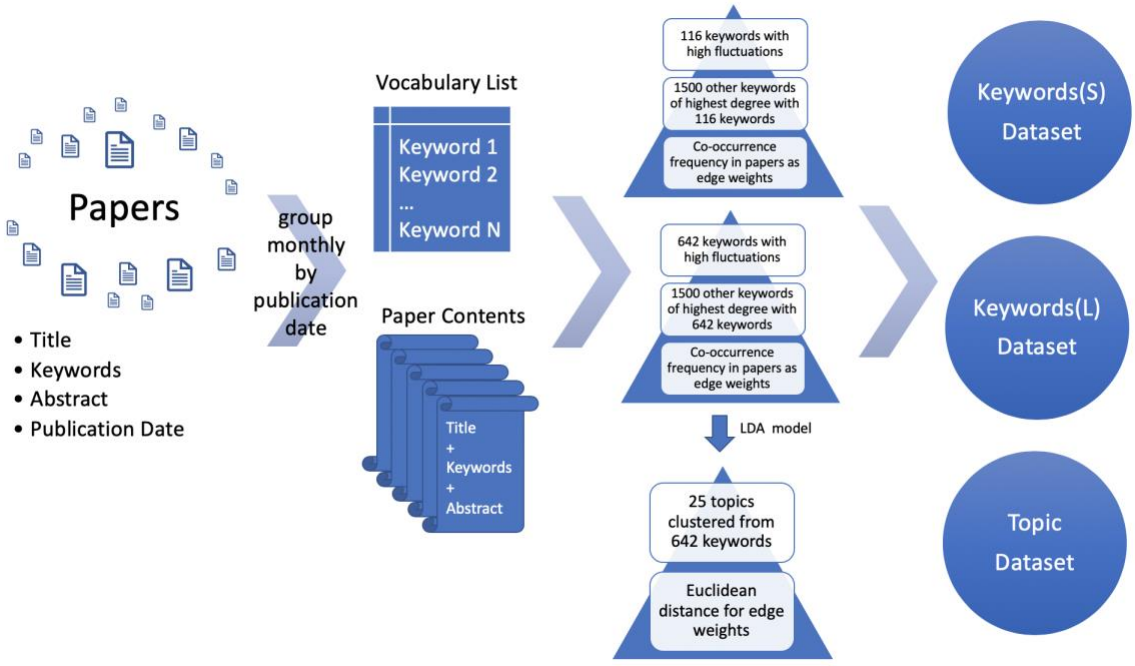


Figure 4 The complete architecture of data preprocessing.

2.4 Data Aggregation Input Pipeline

In each dataset, the data inside can be divided into training set, validation set and testing set, where each example in each set consists of 3 sections – 15 continuous months historical keywords/topics frequency (proportion), accompanying with the graph of last month in 15 months, as well as a certain prediction month. For instance, a training example in Keywords(S) or Keywords(L) dataset consists of 3 parts:

1. Hot keyword frequency proportions from 1st to 15th months, a 15 * 116(642) matrix, as historical data
2. Co-occurrence matrix (adjacency matrix) of all keywords in 15th month, a 1616(2142) * 1616(2142) matrix
3. Hot keyword frequency proportions of 16th (17th, 18th, 19th, 20th) month, a 1 * 116(642) vector, as ground truth

a training example in Topic dataset also consists of 3 parts:

1. Hot topic frequency from 1st to 15th months, a 15 * 25 matrix, as historical data
2. Co-occurrence matrix (adjacency matrix) of topics in 15th month, a 25 * 25 matrix
3. Hot keyword frequency proportions of 16th (17th, 18th, 19th, 20th) month, a 1 * 25 vector, as ground truth

With the explicit explanation of example contents above, the example structure in each dataset becomes clearer. After introducing the generation of each dataset, it comes to the program of data aggregation pipeline. The most popular python framework TensorFlow [34] is adopted to achieve the goal.

2.4.1 TensorFlow Brief Introduction

TensorFlow is an open source python Machine Learning framework serving the model building and deployment purpose, which was first proposed by Google Research in 2016 [34]. TensorFlow provides with various APIs to employ many Machine Learning or Deep Learning algorithms, such as CNN and RNN, and it continues to feed new algorithms in. Also, with the help of its estimators, the training and evaluation process can become easier. Besides the employment of algorithms and estimators, it also offers APIs to build data input pipeline, which is able to handle large amounts of data. Based on the robust framework design, it can also speed up the data loading process during model training, as well as generating batch dataset.

2.4.2 TensorFlow Input Pipeline

An API named tf.data in TensorFlow enables programmers to create input pipelines under TensorFlow framework. It introduces tf.data.Dataset, which can aggregate one or more components in each dataset into a sequence of elements. Taking Keywords(L) dataset for example, an element might be an example in the dataset. Inside each element, there are 3 tensor components representing the hot keyword frequency proportions of 15 months, adjacency matrix of all keywords of 15^{th} month and ground truth of 16^{th} month, respectively.

This input pipeline can aggregate all training examples, validation examples and testing examples into 3 separate files, with file type using a recommended TFRecord format. Every part in each example of different sets is encapsulated into data stream of tensor components as a certain type. The available data stream tensor types are byte, int and float. After defining the tensor type in encoding program file, the decoding program file is capable of restoring the original examples after specifying the data type and data shape in the program.

Since the model used in this project is written with TensorFlow framework, then it is easy to integrate TensorFlow input pipeline into the model. When training the model, the input pipeline is able to generate batches of training data into the model with the designed batch size, as well as batches of validation data to validate the training performance. While in testing stage, this input pipeline is also able to provide with batches of the testing examples. The whole process is illustrated below, see Figure 5.

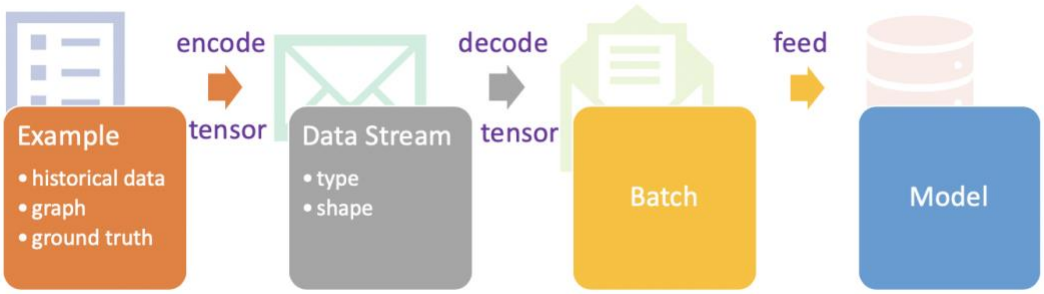


Figure 5 The process of TensorFlow input pipeline.

Chapter 3 Methods

3.1 Graph Neural Network Introduction

Graph Neural Network was first proposed by Marco Gori in 2005 [35], and the improvement of its theoretical foundation was made by Franco Scarselli in 2009 [36]. In the early studies of GNN, researchers fall into the category of recurrent graph neural networks (RecGNNs). They learn a target node's representation by propagating neighbor information in an iterative manner until a stable fixed point is reached [13]. This process is computationally expensive, so other researchers are devoting themselves to overcoming the challenges. Meanwhile, with the fast development of CNN in Euclidean domain, some researchers are trying to re-define the convolution operation on graph structured data.

Unlike traditional neural networks taking data from Euclidean domain as input, such as audios and images, GNN takes graphs as input, which comes from non-Euclidean domain. To be more specific, traditional neural networks take grid-structured data as input, while GNN takes irregular graph-structured data as input, see the comparison in Figure 6. At first, GNN takes only proper graph-structured data as input, such as molecular structure, however, many scenarios in the real world can be converted to graphs, such as road maps. Based on the above generalization of graphs, GNN could be widely applied to many different academic domains.

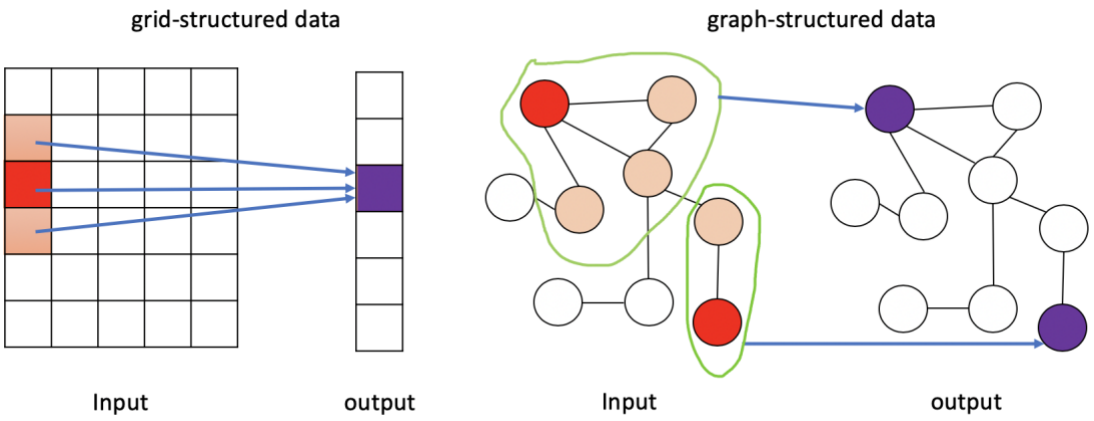


Figure 6 Comparison between Euclidean (grid-structured) data and non-Euclidean (graph-structured) data, and the difference of conducting convolutions.

Each node in grid-structured data has fixed-size of neighbors and the neighbors are arranged in the similar orders, while nodes in graphs are connected with different size of neighbors and the neighbors are arranged in diverse orders. It is easier to collect neighbor information from data in Euclidean domain based on the same kernel structure during convolution, while it is hard to collect neighbor information in non-Euclidean domain, due to the various size of neighbors connected with each node in the graph, see the different convolution process in Figure 6. Thus, the research related to graph convolution aims to solve the above problem, where Graph Convolution Networks are proposed.

3.1.1 Graph Convolution Networks Theory

In 2013, Bruna et al. [37] proposed a graph convolution method based on the research of graph signal processing, conducted by Shuman et al. [38]. In Bruna's paper, he proposed 2 ways to conduct graph convolution – one is from spatial domain and the other one is from spectral domain. Convolution on spatial domain is a quite

straightforward method, it takes all the neighbors of each node based on a certain algorithm and processes the neighbor information accordingly, which is much more similar to conducting convolution in Euclidean domain. Since convolution on spatial domain is not the core foundation of model used in this paper, there is no further elaboration. If interested, the detailed explanation can be found in paper proposed by Niepert et al. [39].

Convolution on spectral domain is based on spectral decomposition (eigen decomposition) of Laplacian matrix. Laplacian matrix L is defined as,

$$L = D - A \tag{4}$$

where D and A are the degree matrix and adjacency matrix of an undirected graph, respectively. See Figure 7 for detailed illustration.

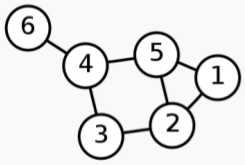
Labelled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

Figure 7 Laplacian matrix of a labelled, undirected graph [40].

There are 3 different forms of Laplacian matrix, and Equation (4) is called combinatorial Laplacian matrix, which is for simple graphs like Figure 7. The second form is called random walk normalized Laplacian matrix, which is defined as,

$$L^{rw} = D^{-1}L \tag{5}$$

where D is still the degree matrix of Laplacian matrix.

The third form, symmetric normalized Laplacian matrix, is the form used in eigen decomposition of Laplacian matrix, defined as,

$$L^{sys} = D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I^n - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \quad (6)$$

where D and A are still the degree matrix and adjacency matrix of an undirected graph, respectively. Laplacian matrix is Semi-Positive Definite Symmetric Matrix, so there are 3 properties which are useful in graph convolution:

1. Laplacian matrix as a symmetric matrix has n linearly independent eigenvectors
2. Laplacian matrix as Semi-Positive Definite Matrix has non-negative eigenvalues
3. Laplacian matrix as a symmetric matrix has orthogonal eigenvectors, which means the matrix formed by eigenvectors is an orthogonal matrix

Not all matrices can do eigen decomposition, but only n_{th} order square matrices with n linearly independent eigenvectors. Based on the above properties, Laplacian matrix is able to perform eigen decomposition as,

$$L = U\Lambda U^{-1} \quad (7)$$

where U is the matrix of eigenvectors of L, Λ is the diagonal degree matrix of eigenvalues of Laplacian matrix. Due to U is an orthogonal matrix, then $UU^T = E$. With definition of inverse matrix, it is proven $UU^{-1} = E$. Therefore, $U^{-1} = U^T$ in Laplacian matrix, then Equation (7) can be re-written as,

$$L = U\Lambda U^T \quad (8)$$

Since the graph convolution is conducted in spectral domain, it is necessary to

transform input of the graph convolution layer into spectral domain. Thus, Fourier transform is needed to be applied. The details of why Fourier transform is able to be extended into graph structures is available in paper proposed by Shuman et al. [38]. Based on the research from Shuman's paper, the linearly independent orthogonal vectors of Laplacian matrix can be regarded as Fourier base, while the eigenvalues can be regarded as corresponding frequencies. Therefore, the input of the graph convolution layer is able to be transformed into spectral domain. After transforming into spectral domain, graph convolution is able to be employed on the input based on Laplacian matrix.

Considering x as the input of graph convolution layer, then Fourier transform is conducted on x by $U^T x$. The diagonal eigenvalue matrix Λ of Laplacian matrix can be written as,

$$\begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}$$

where λ_1 to λ_n are n eigenvalues of Laplacian matrix. A convolution kernel h is designed based on eigenvalue matrix Λ, and Fourier transformed version $\hat{h}(\lambda_l), l \in \{1, \dots, n\}$, is defined here, written as,

$$\begin{pmatrix} \hat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n) \end{pmatrix}$$

then the Fourier transformed product between convolution kernel $\hat{h}$ and input x becomes $\hat{h}U^T x$. After applying the inverse transformation U on Fourier transformed product, the graph convolution can be described as $U\hat{h}U^T x$. In a more general way, the equation is

written as,

$$(x * h)_G = U((U^T h) \odot (U^T x)) \tag{9}$$

where x is the input of graph convolution layer, h is the graph convolution kernel, $\odot$ indicates Hadamard product, and U^T is the Fourier base.

Graph convolution collects neighbor information based on hops. For example, a node itself is the first hop, which denotes the information of the node itself, then all the direct neighbors connected with this node is the second hop (see nodes colored shallow orange in Figure 7 on the right), etc. When mapping hop definition to graph convolution kernel, eigenvalues of Laplacian matrix is considered as the designed hops to collect neighbor information of a certain node within the number of hops.

While applying convolution inside neural networks, the trainable kernels are required, which can also be locally shared. The most straightforward method is to turn $\hat{h}(\lambda_l), l \in \{1, \dots, n\}$ as convolution kernel $\theta(\Lambda)$, then the output of graph convolution can be written as,

$$y_{output} = \sigma\,(U\theta(\Lambda)U^T x) \tag{10}$$

where σ is the activation function, and x is the input. However, the computation cost is too expensive when utilizing the forward propagation, and in this way, the convolution kernel $\theta(\Lambda)$ is not capable of sharing locally.

Based on the above consideration, Chebyshev Polynomials Approximation is introduced to reduce the computational cost [41]. $\hat{h}(\lambda_l), l \in \{1, \dots, n\}$ can be well-approximated by a truncated expansion in terms of Chebyshev polynomials up to Kth order, with products between trainable parameters $\{\alpha_1, \alpha_2, \dots, \alpha_k\}$ and eigenvalues. The

trainable parameters can be initialized randomly and adjusted during backpropagation. Also, after introducing the above trainable parameters, the eigen decomposition of Laplacian matrix is not necessary during forward propagation. Instead, the Laplacian matrices of each hop is calculated. Thus, the output of graph convolution can be written as,

$$y_{output} = \sigma\left(\sum_{i=0}^{K} \alpha_i L^i x\right) \tag{11}$$

where σ is the activation function, α_i is the ith trainable parameter, L^i is Laplacian matrix for i-hop neighbors of a certain node and x is the input from last layer. If Chebyshev Polynomials Approximation is restricted to 2nd order, it becomes the form in the paper proposed by Kipf and Welling [42].

Based on the above modification, the current graph convolution kernel has good spatial localization, and K is the receptive field of graph convolution kernel. During each convolution, the weighted sum of K-hop neighbors is calculated for every node to aggregate the neighbors feature.

3.1.2 Spatio-Temporal Graph Network Introduction

Just as mentioned in Introduction section of this paper, we adopted the spatial temporal graph neural network to be our model. The model used in this paper is inspired by the model proposed by Yu et al. [21]. They proposed a spatio-temporal graph convolution networks, named STGCN. There are 2 main parts of STGCN – spatio-temporal convolutional block (ST-conv Block) and output layer.

Inside ST-conv Block, there is a “sandwich” structure to capture temporal and spatial

information, which is described as one temporal convolution layer + one spatial convolution layer + another temporal convolution layer. Temporal information is captured by 1-d causal convolution, followed by Gated Linear Units in order to add non-linearity. Causal convolution is a variety of CNN, which is dealing with sequential data, such as the time-series frequencies of a certain node in our dataset. The feature of 1-d causal convolution is that it takes time series into consideration, therefore, at time t, only input before time t can be seen in the model training process. The mechanism is illustrated in Figure 8 below. Gated Linear Unit (GLU) [43] introduces gate mechanism into traditional CNN, which can reduce the problem of gradients vanishing while keeping non-linearity. GLU is defined as,

$$h_l(X) = (X * W + b) \otimes \sigma(X * V + c) \tag{12}$$

where X is the input of layer h_l, W, V, b and c are learned parameters, σ is sigmoid function, and $\otimes$ is element-wise product between matrices. Spatial information is captured by Graph Convolution Network, and the theory of GCN is introduced in section 3.1.1.

Between 2 ST-conv Blocks, there is an operation called Layer Normalization [44]. Literally, Layer Normalization is normalizing the neurons inside a certain layer. It takes all input dimensions inside the corresponding layer, and normalize the inputs based on their mean and standard deviation, calculated as,

$$\mu = \sum_i x_i \ , \sigma = \sqrt{\sum_i (x_i - \mu)^2 + \varepsilon} \tag{13}$$

where μ is mean of all neurons, σ is standard deviation, x_i means ith neuron inside the layer, and ε is a very small number to avoid the divisor to be 0. The above calculation is based on all neurons inside the layer, with no relations to the batch examples fed into the

layer. After obtaining the mean and standard deviation of all inputs, the normalization is calculated as,

$$y_i = \frac{x_i - \mu}{\sigma^2} \tag{14}$$

where y_i is output of the layer, other parameters have same meanings as Equation (13).

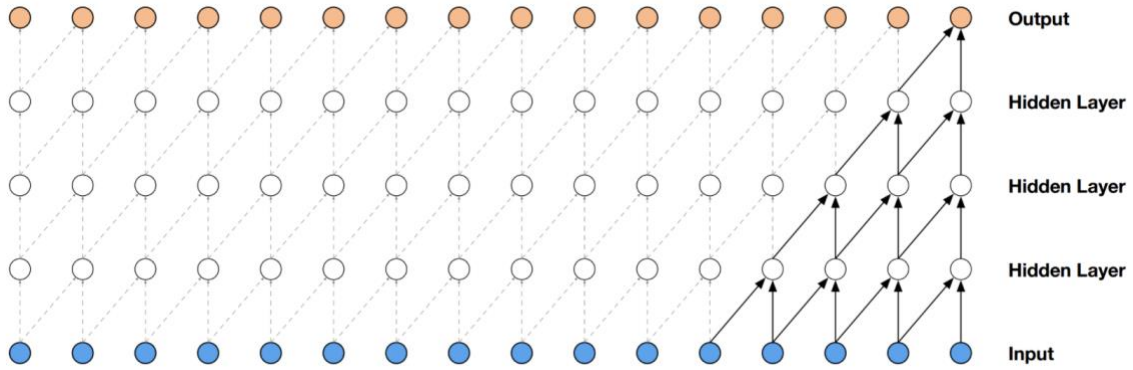


Figure 8 Illustration of the mechanism in causal convolution [45].

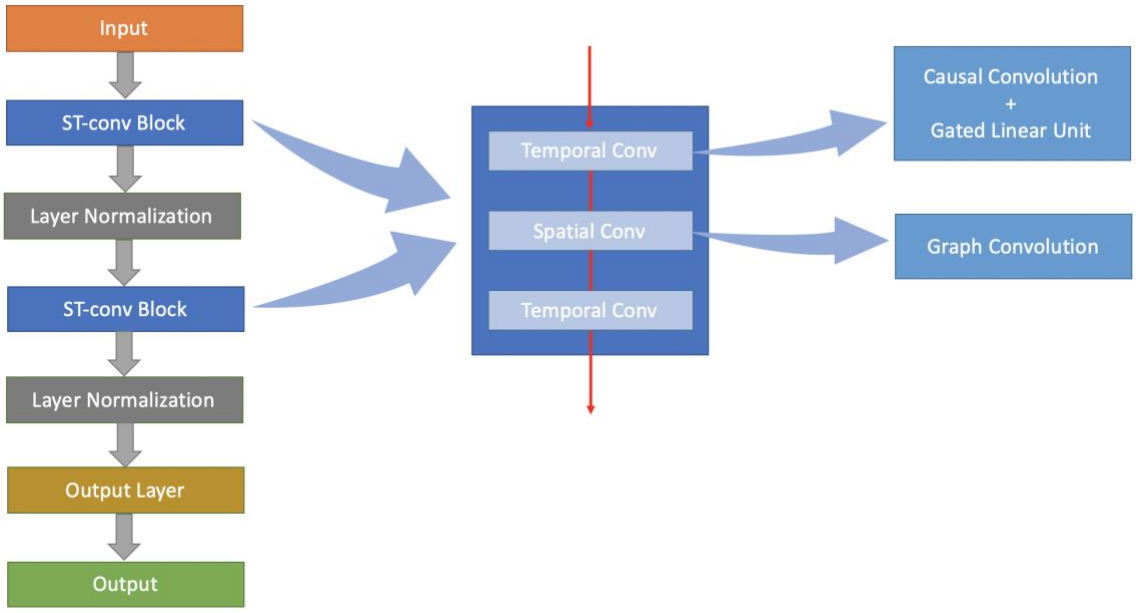


Figure 9 STGCN model architecture.

Inside the output layer, there are another 2 temporal layers to capture the temporal feature, with sigmoid function as activation function, then integrate features to be the prediction output. The architecture of STGCN is illustrated in Figure 9 above, with ST-conv Block used twice in their paper.

3.2 Dynamic Spatio-Temporal Graph Network Model

Inspired by STGCN model introduced above, we modified the spatial convolution layer inside ST-conv Block. STGCN model only takes static graph as input to spatial convolution layer, i.e. there is only one graph loaded into the model. However, the model cannot be tailored to fit into the datasets in this project. In each dataset of this project, the graphs vary from month to month. Based on the above demand, we modified the static spatial convolution layer into a dynamic one. The comparison shown in ST-conv Block between the spatial convolution layers is demonstrated in Figure 11 (original static model) and Figure 12 (modified dynamic model) below.

From Figure 11 and Figure 12, it is quite clear that our modified model can take dynamic graphs as input and combine each graph information into corresponding temporal information from different input examples. Inside the modified model, each GCN element is grouped with an adjacency matrix of a graph. The Laplacian matrix is different in every GCN element, which means GCN layer can collect different graph information from different input examples. Therefore, more dynamic graph information can be fed during model training.

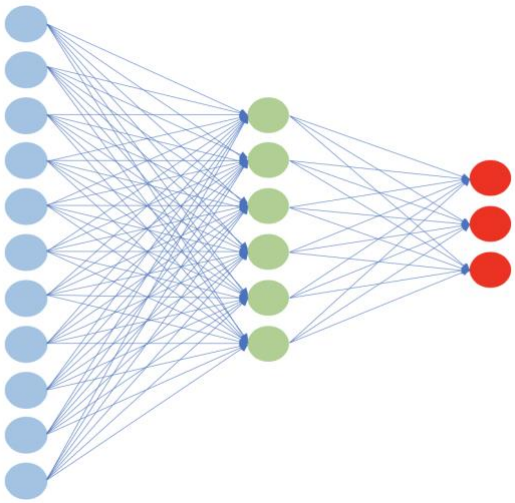


Figure 10 Demonstration of fully connected layer.

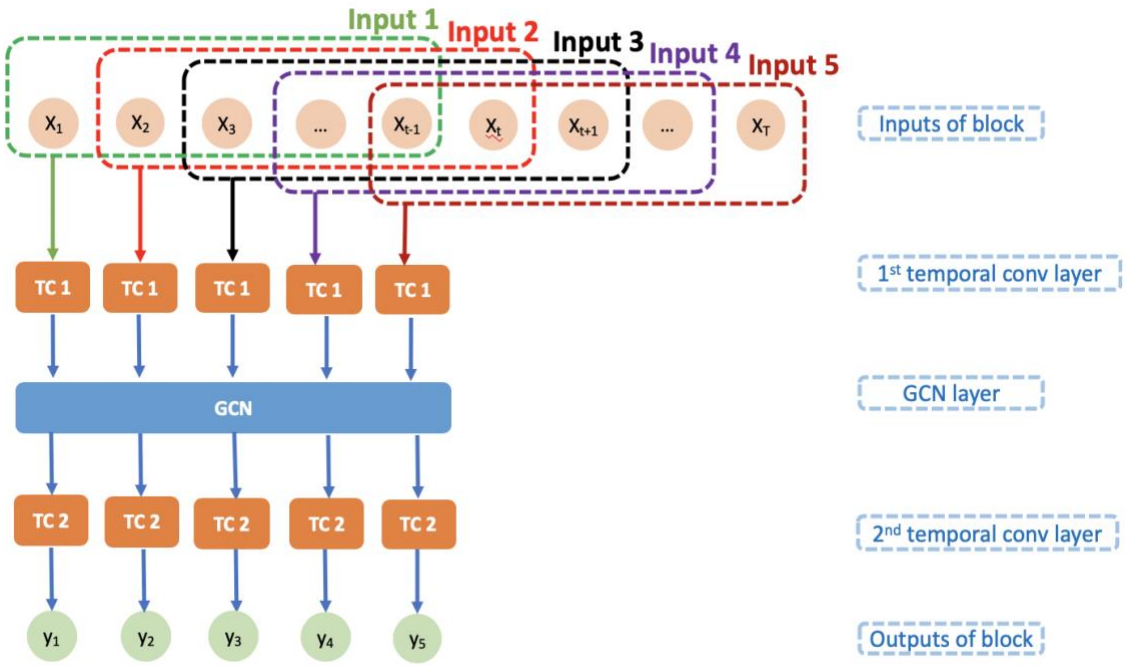


Figure 11 Original static model.

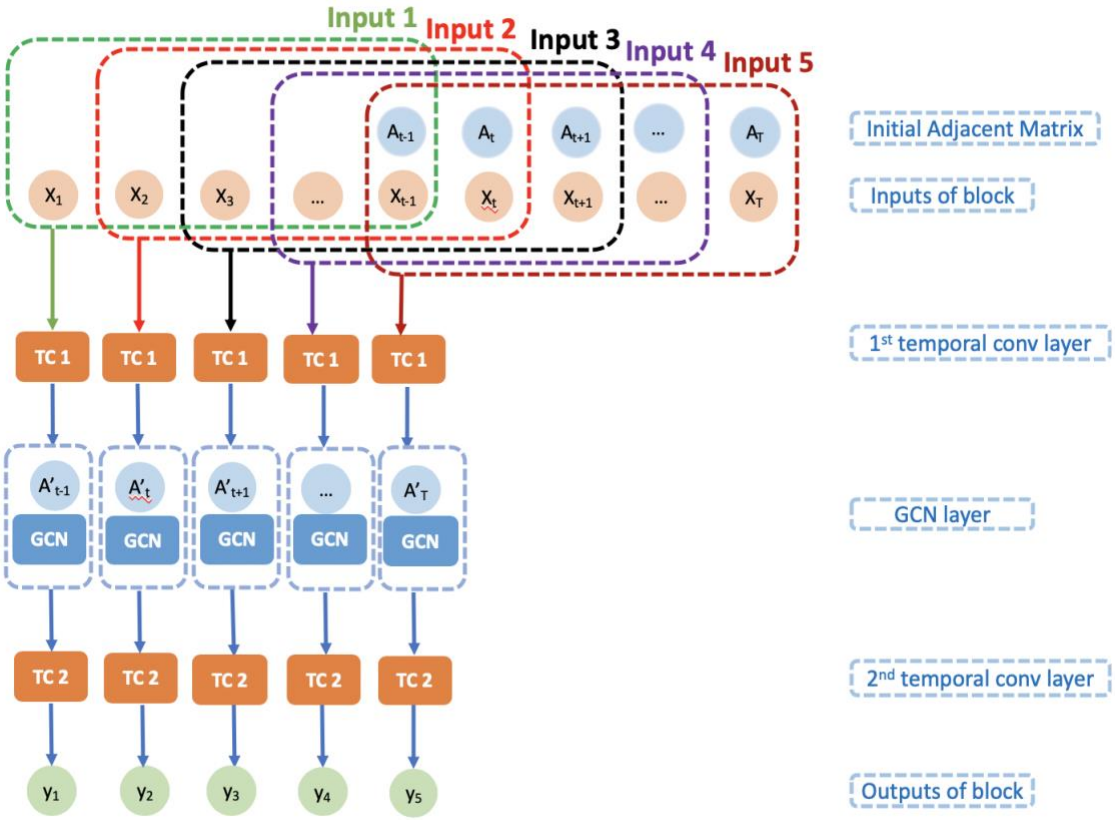


Figure 12 Modified dynamic model.

Other than the modification in GCN layer, a fully connected layer is added after output layer of the original model. Therefore, more feature representations can be integrated from all nodes in the model. See Figure 10 for demonstration of fully connected layer. In addition, since there is longer temporal input data in our dataset, then one more ST-conv Block is employed in our final model, to collect more higher-level temporal information. See Figure 13 for the whole modified model.

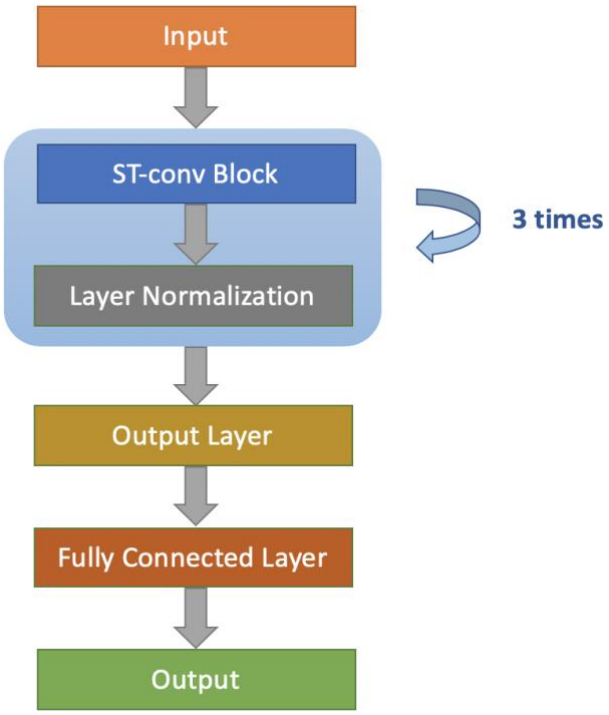


Figure 13 Whole modified model.

3.3 Model Implementation

TensorFlow is also adopted for the model implementation. TensorFlow is able to utilize GPU as an accelerating option, so all the experiments are run on a server with GPU. The detailed experiment settings are available in Chapter 4. In every program constructed with TensorFlow, it always contains 2 parts – one is building the computation graph, the other one is executing the graph inside a session. The computation graph consists of Tensor and Operation – Tensor can be considered as a vector or a multi-dimensional matrix based on the data assigned, and Operation literally means an operation conducted among different data, such as addition and multiplication. The construction of computation graph is executed before feeding the input data, so it is

necessary to pre-define a placeholder of the data shape.

In this project, the model implementation is completely based on the above procedure. The construction of computation graph follows the model presented in Figure 13, with the APIs provided in tf.nn module, such as tf.nn.conv2d and tf.nn.sigmoid, etc. The input data is constructed with TensorFlow data input pipelines, which is introduced in Chapter 2.

Chapter 4 Experiment Results

4.1 Experiment Settings

All the experiments are conducted on a Linux server – CPU: Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz, GPU: NVIDIA GeForce GTX 1080. In all test datasets, the historical data is formed by 15 months, then used to predict data in the next 5 months with separate models, i.e. data of first 15 months in each example is used to predict keyword/topic frequency (proportions) of 16th, 17th, 18th, 19th, 20th month with different models.

4.2 Result Evaluation Benchmark

Results in each dataset are illustrated as 3 parts with comparison to the direct copy of corresponding keyword/topic frequency (proportion) in last one month (15th month) of historical 15 months, as evaluation benchmark. For example, if keyword/topic frequencies in 17th month are predicted, then data in 15th month ground truth is serving as evaluation benchmark. The 3 parts of results are:

1. Pearson correlation coefficient between prediction and ground truth, indicating the linear relationship between prediction and ground truth
2. Hit accuracy of top fluctuated keywords/topics, indicating how many fluctuated keywords/topics are able to be predicted in top 5, top 10 and top 20 of prediction results
3. Prediction in continuous time series, indicating the trend of specific hot

keywords/topics

See paragraphs below for the explanation of Pearson correlation coefficient, top fluctuated keywords/topics, hit accuracy, and prediction in continuous time series, where hit accuracy is calculated based on the definition of top fluctuated keywords/topics.

Pearson correlation coefficient is a statistical linear correlation measurement of 2 variables (prediction or ground truth vectors in this paper), with the value between -1 and 1. -1 and 1 mean negative linear correlation and positive correlation, respectively, while 0 means there is no linear correlation between 2 variables. Pearson correlation coefficient $\rho_{X,Y}$ is defined as the covariance between 2 variables divided by the standard deviations, calculated as,

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{15}$$

where $cov(X,Y)$ is covariance between X and Y, σ_X and σ_Y are the standard deviations of X and Y, respectively. There is no difference if 2 variables change the position during the calculation, which means $\rho_{X,Y}$ and $\rho_{Y,X}$ are the same. In each dataset result, the prediction or ground truth of all hot keywords/topics can be treated as a vector. Pearson correlation coefficient is calculated in each testing example between prediction vector and corresponding ground truth vector, as well as the ground truth vector of prediction month and ground truth vector of last month in historical data, serving for evaluation benchmark. For example, if keyword/topic frequencies in 17th month are predicted, then Pearson correlation coefficient between 17th month prediction and 17th month ground truth is calculated, as well as 17th month ground truth and 15th month ground truth, with

latter serving as evaluation benchmark. In section 4.3 and 4.4, all results of Pearson correlation coefficient are illustrated in 11 testing examples, indexed from 0 to 10, along with an average calculated based on 11 examples. Also, all the results are sorted based on Pearson correlation coefficients of ith month ground truth and 15th month ground truth (benchmark), where i is selected from {16, 17, 18, 19, 20}.

The top 5/10/20 fluctuated keywords/topics in each testing example are selected based on the top 5/10/20 highest ratio inside vector r defined as,

$$r = \frac{(frequency_{\ i^{th}\ month} - frequency_{ground\ truth\ 15^{th}})}{frequency_{ground\ truth\ 15^{th}}} \quad (16)$$

where $frequency_{ground\ truth\ 15^{th}}$ is the vector of all hot keyword/topic frequency (proportion) of 15th month in historical data, and $frequency_{\ i^{th}\ month}$ is the vector of ground truth or prediction frequency in the prediction month (16th, 17th, 18th, 19th or 20th month).

Hit accuracy is defined based on fluctuated keywords/topics. Similar to evaluations on Pearson correlation coefficient, hit accuracy is also compared between hot keyword/topic frequency of last month in historical data and in prediction month. The hit accuracy in each dataset is illustrated as a table (see results in section 4.3, 4.4), similar to confusion matrix. Prediction top 5/10/20 means the top 5/10/20 fluctuated keywords/topics in prediction vector, with prediction vector selected as $frequency_{\ i^{th}\ month}$ to calculate r in Equation 16, while Ground Truth top 5/10/20 means the top 5/10/20 fluctuated keywords/topics in ground truth vector of prediction month, with ground truth vector in prediction month selected as $frequency_{\ i^{th}\ month}$ to calculate r in Equation 16. The value in each table is considered as the intersection

between Prediction top 5/10/20 and Ground Truth top 5/10/20. For example, the value in row Ground Truth top 5 and column Prediction top 10 means how many keywords/topics are both in Ground Truth top 5 and Prediction top 10, which indicates the how many top 5 fluctuated keywords/topics are able to be predicted in prediction top 10 fluctuated keywords/topics. There are 3 tables selected in each dataset, with best, worst and average hit accuracy, respectively.

Prediction in continuous time series is performing in the same datasets, but without random sample. Therefore, no matter training set, validation set, or testing set in each dataset is formed in time series. Utilizing the model trained with 15 months historical data to predict data in 16th month, the outcome of all testing examples can form the predictions in continuous time series of every hot keyword/topic. The figures listed in section 4.3 and 4.4 are randomly selected. The x axis indicates the month in time series, while y axis indicates the normalized word frequency (proportion). The number listed on the top of each figure is the index of the selected keyword/topic, and ground truth is listed with red line, while prediction is listed with blue line.

4.3 Keyword Dataset Result

4.3.1 Keywords(S) Dataset

Pearson correlation coefficient

1. 16th month

Table 1 Keywords(S) Dataset 16th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	16th Ground Truth & 15th Ground Truth	16th Ground Truth & 16th Prediction

0	0.672	0.844
1	0.732	0.881
2	0.733	0.797
3	0.821	0.878
4	0.884	0.80
5	0.889	0.874
6	0.911	0.915
7	0.944	0.963
8	0.945	0.835
9	0.949	0.953
10	0.962	0.964
Average	0.859	0.882

2. 17th month

Table 2 Keywords(S) Dataset 17th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	17th Ground Truth & 15th Ground Truth	17th Ground Truth & 17th Prediction
0	0.638	0.874
1	0.762	0.800
2	0.782	0.898
3	0.846	0.916
4	0.865	0.945
5	0.884	0.899
6	0.914	0.914
7	0.945	0.916
8	0.955	0.938
9	0.958	0.962
10	0.966	0.933
Average	0.865	0.909

3. 18th month

Table 3 Keywords(S) Dataset 18th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	18th Ground Truth & 15th Ground Truth	18th Ground Truth & 18th Prediction
0	0.783	0.837
1	0.848	0.893
2	0.856	0.843
3	0.858	0.864
4	0.892	0.915
5	0.934	0.920
6	0.935	0.941
7	0.946	0.942
8	0.952	0.924
9	0.956	0.932
10	0.970	0.941
Average	0.903	0.905

4. 19th month

Table 4 Keywords(S) Dataset 19th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	19th Ground Truth & 15th Ground Truth	19th Ground Truth & 19th Prediction
0	0.731	0.850
1	0.814	0.944
2	0.845	0.901
3	0.876	0.894
4	0.878	0.855
5	0.880	0.914
6	0.898	0.915
7	0.908	0.900
8	0.916	0.903
9	0.946	0.931
10	0.958	0.927
Average	0.877	0.902

5. 20th month

Table 5 Keywords(S) Dataset 20th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	20th Ground Truth & 15th Ground Truth	20th Ground Truth & 20th Prediction
0	0.701	0.892
1	0.725	0.836
2	0.740	0.685
3	0.836	0.906
4	0.856	0.886
5	0.897	0.859
6	0.903	0.917
7	0.924	0.932
8	0.940	0.924
9	0.948	0.942
10	0.957	0.929
Average	0.857	0.883

Hit Accuracy

1. 16th month

Table 6 Keywords(S) Dataset 16th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	4	5	5
Ground Truth top 10	5	9	10
Ground Truth top 20	5	10	18

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	1	1
Ground Truth top 10	1	1	1
Ground Truth top 20	1	1	2

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	2	5
Ground Truth top 10	3	4	7
Ground Truth top 20	3	4	8

2. 17th month

Table 7 Keywords(S) Dataset 17th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	2	3	4
Ground Truth top 10	4	7	8
Ground Truth top 20	5	8	13

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	0	0	1
Ground Truth top 10	0	0	1
Ground Truth top 20	0	0	2

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	0	2	3
Ground Truth top 10	1	4	5
Ground Truth top 20	1	5	7

3. 18th month

Table 8 Keywords(S) Dataset 18th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	4	5	5
Ground Truth top 10	5	7	10

Ground Truth top 20	5	8	15

Worst hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	0	1	1
Ground Truth top 10	0	2	3
Ground Truth top 20	0	2	4

Average hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	1	1	2
Ground Truth top 10	2	3	4
Ground Truth top 20	4	7	9

4. 19th month

Table 9 Keywords(S) Dataset 19th month Hit accuracy

Best hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	1	2	4
Ground Truth top 10	3	4	8
Ground Truth top 20	5	9	16

Worst hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	0	0	0
Ground Truth top 10	0	0	1
Ground Truth top 20	0	1	2

Average hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	4	4	5

Ground Truth top 10	4	8	9
Ground Truth top 20	4	9	12

5. 20th month

Table 10 Keywords(S) Dataset 20th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	2	3	3
Ground Truth top 10	3	5	8
Ground Truth top 20	4	9	16

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	1	1
Ground Truth top 10	1	2	2
Ground Truth top 20	2	3	3

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	2	3	5
Ground Truth top 10	3	5	8
Ground Truth top 20	4	7	11

Prediction of Specific Keywords in Time series

Ground truth: red line, Prediction: blue line

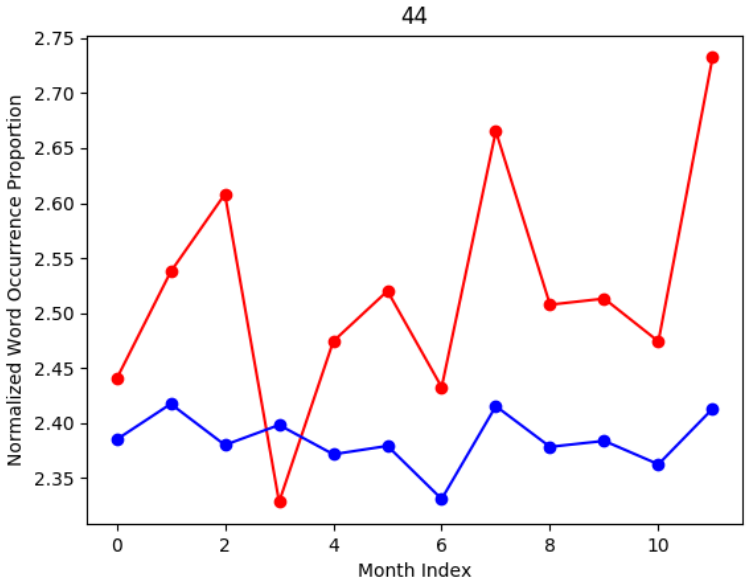


Figure 14 Keywords(S) Dataset predictions of specific keywords – keyword index 44

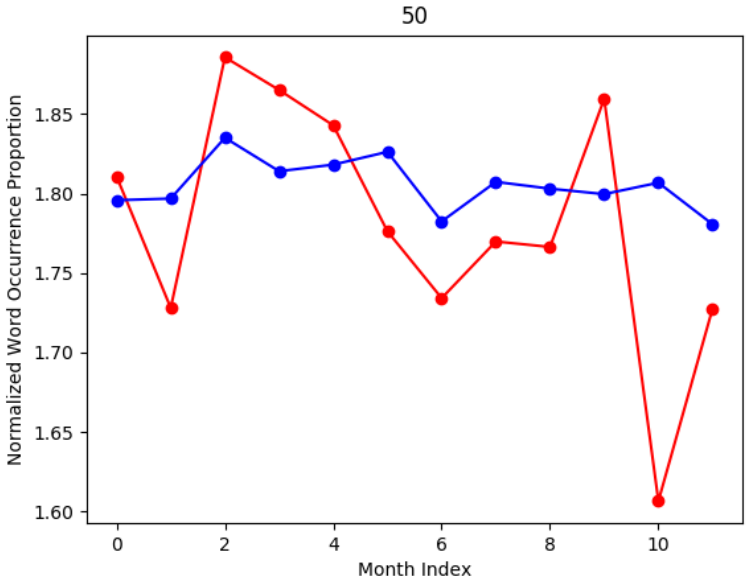


Figure 15 Keywords(S) Dataset predictions of specific keywords – keyword index 50

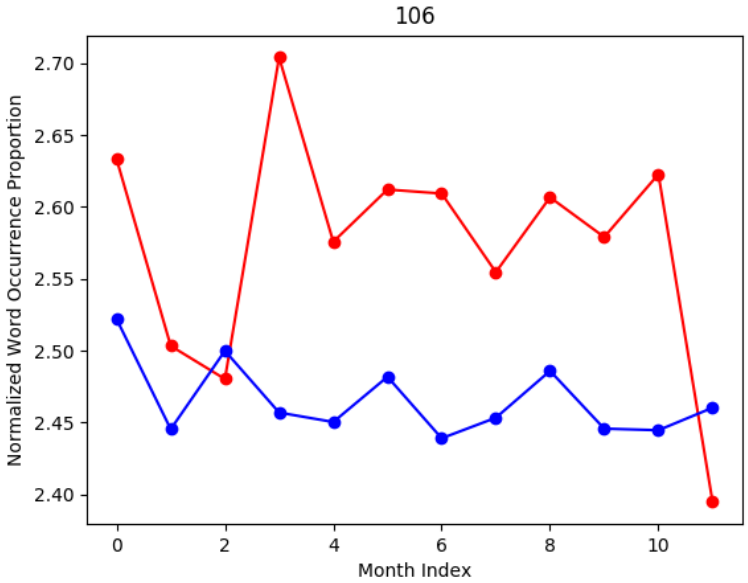


Figure 16 Keywords(S) Dataset predictions of specific keywords – keyword index 106

4.3.2 Keywords(L) Dataset

Pearson correlation coefficient

1. 16th month

Table 11 Keywords(L) Dataset 16th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	16th Ground Truth & 15th Ground Truth	16th Ground Truth & 16th Prediction
0	0.959	0.970
1	0.978	0.980
2	0.979	0.979
3	0.984	0.987
4	0.984	0.986
5	0.985	0.988
6	0.995	0.993
7	0.996	0.993
8	0.996	0.995
9	0.996	0.993

10	0.996	0.996
Average	0.986	0.987

2. 17th month

Table 12 Keywords(L) Dataset 17th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	17th Ground Truth & 15th Ground Truth	17th Ground Truth & 17th Prediction
0	0.903	0.960
1	0.964	0.965
2	0.988	0.987
3	0.988	0.990
4	0.989	0.986
5	0.990	0.987
6	0.994	0.993
7	0.994	0.994
8	0.994	0.996
9	0.995	0.994
10	0.997	0.995
Average	0.981	0.986

3. 18th month

Table 13 Keywords(L) Dataset 18th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	18th Ground Truth & 15th Ground Truth	18th Ground Truth & 18th Prediction
0	0.913	0.945
1	0.957	0.957
2	0.966	0.960
3	0.970	0.985
4	0.979	0.976
5	0.984	0.990
6	0.987	0.990
7	0.992	0.992

8	0.995	0.991
9	0.995	0.988
10	0.997	0.992
Average	0.976	0.979

4. 19th month

Table 14 Keywords(L) Dataset 19th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	19th Ground Truth & 15th Ground Truth	19th Ground Truth & 19th Prediction
0	0.981	0.988
1	0.985	0.991
2	0.985	0.988
3	0.987	0.990
4	0.992	0.995
5	0.992	0.994
6	0.994	0.994
7	0.995	0.993
8	0.996	0.998
9	0.997	0.991
10	0.997	0.996
Average	0.991	0.992

5. 20th month

Table 15 Keywords(L) Dataset 20th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	20th Ground Truth & 15th Ground Truth	20th Ground Truth & 20th Prediction
0	0.960	0.971
1	0.983	0.988
2	0.985	0.981
3	0.987	0.986
4	0.987	0.990
5	0.989	0.994

6	0.992	0.991
7	0.996	0.989
8	0.996	0.995
9	0.996	0.984
10	0.996	0.987
Average	0.987	0.987

Hit Accuracy

1. 16th month

Table 16 Keywords(L) Dataset 16th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	3	4	5
Ground Truth top 10	5	8	9
Ground Truth top 20	5	10	16

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	0	1	1
Ground Truth top 10	0	1	1
Ground Truth top 20	0	1	1

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	3	5
Ground Truth top 10	1	5	10
Ground Truth top 20	1	5	10

2. 17th month

Table 17 Keywords(L) Dataset 17th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	2	5	5
Ground Truth top 10	5	9	10
Ground Truth top 20	5	10	16

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	1	1
Ground Truth top 10	1	2	3
Ground Truth top 20	1	3	5

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	1	4
Ground Truth top 10	1	3	7
Ground Truth top 20	2	4	9

3. 18th month

Table 18 Keywords(L) Dataset 18th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	3	4	5
Ground Truth top 10	5	7	10
Ground Truth top 20	5	10	17

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	0	0	0
Ground Truth top 10	0	0	0
Ground Truth top 20	1	1	2

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	4	4	4
Ground Truth top 10	4	4	4
Ground Truth top 20	4	5	6

4. 19th month

Table 19 Keywords(L) Dataset 19th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	3	5	5
Ground Truth top 10	5	8	9
Ground Truth top 20	5	10	18

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	3	3	3
Ground Truth top 10	3	3	3
Ground Truth top 20	3	3	3

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	2	4	5
Ground Truth top 10	2	6	8
Ground Truth top 20	2	6	10

5. 20th month

Table 20 Keywords(L) Dataset 20th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20

Ground Truth top 5	4	5	5
Ground Truth top 10	5	7	10
Ground Truth top 20	5	10	13

Worst hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	0	0	2
Ground Truth top 10	0	0	2
Ground Truth top 20	0	0	2

Average hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	0	1	1
Ground Truth top 10	1	1	3
Ground Truth top 20	3	5	11

Prediction of Specific Keywords in Time series

Ground truth: red line, Prediction: blue line

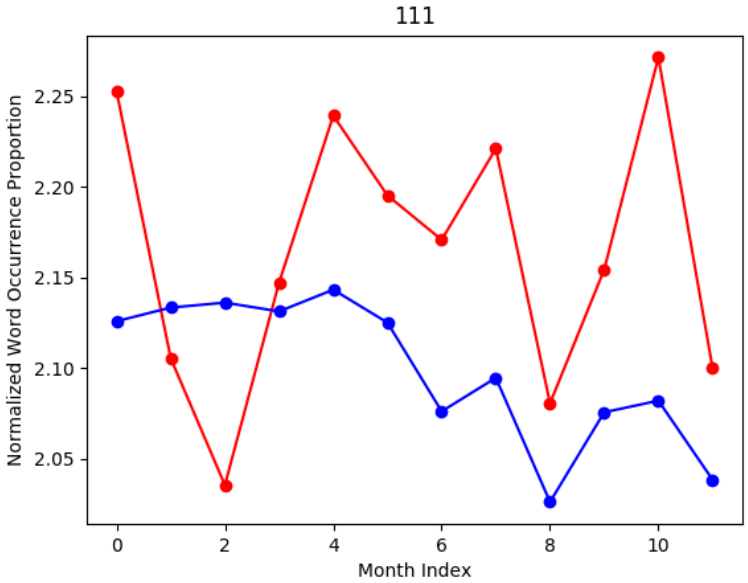


Figure 17 Keywords(L) Dataset predictions of specific keywords – keyword index 111

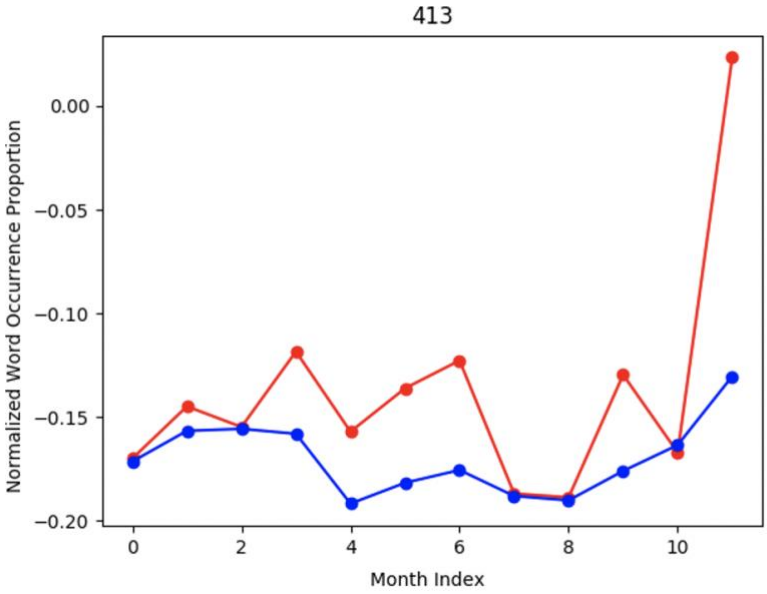


Figure 18 Keywords(L) Dataset predictions of specific keywords – keyword index 413

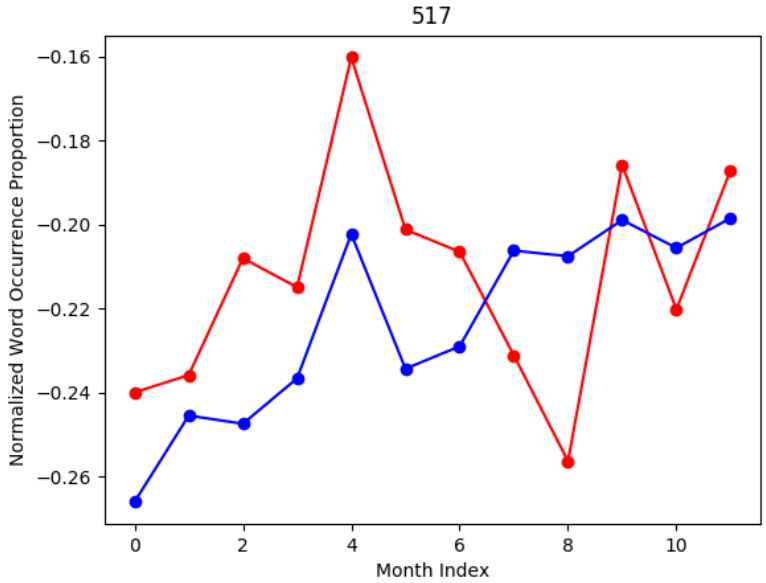


Figure 19 Keywords(L) Dataset predictions of specific keywords – keyword index 517

4.4 Topic Dataset Result

Pearson correlation coefficient

1. 16th month

Table 21 Topic Dataset 16th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	16th Ground Truth & 15th Ground Truth	16th Ground Truth & 16th Prediction
0	0.834	0.913
1	0.983	0.993
2	0.991	0.962
3	0.995	0.998
4	0.995	0.998
5	0.997	0.997
6	0.997	0.994
7	0.998	0.999
8	0.998	0.999
9	0.999	0.999

10	0.999	0.999
Average	0.980	0.986

2. 17th month

Table 22 Topic Dataset 17th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	17th Ground Truth & 15th Ground Truth	17th Ground Truth & 17th Prediction
0	0.894	0.993
1	0.945	0.991
2	0.965	0.989
3	0.980	0.985
4	0.988	0.992
5	0.989	0.990
6	0.990	0.983
7	0.993	0.996
8	0.993	0.998
9	0.998	0.997
10	0.999	0.999
Average	0.976	0.992

3. 18th month

Table 23 Topic Dataset 18th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	18th Ground Truth & 15th Ground Truth	18th Ground Truth & 18th Prediction
0	0.863	0.989
1	0.907	0.991
2	0.937	0.910
3	0.987	0.981
4	0.991	0.984
5	0.991	0.988
6	0.995	0.995
7	0.996	0.985

8	0.997	0.998
9	0.998	0.990
10	0.999	0.994
Average	0.969	0.982

4. 19th month

Table 24 Topic Dataset 19th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	19th Ground Truth & 15th Ground Truth	19th Ground Truth & 19th Prediction
0	0.888	0.974
1	0.968	0.965
2	0.977	0.976
3	0.987	0.996
4	0.992	0.996
5	0.996	0.998
6	0.997	0.981
7	0.997	0.993
8	0.997	0.995
9	0.998	0.999
10	0.999	0.999
Average	0.981	0.988

5. 20th month

Table 25 Topic Dataset 20th month Pearson correlation coefficient

Testing Example Index	Pearson correlation coefficient	
	20th Ground Truth & 15th Ground Truth	20th Ground Truth & 20th Prediction
0	0.895	0.978
1	0.934	0.942
2	0.964	0.984
3	0.974	0.976
4	0.981	0.983
5	0.990	0.997

6	0.993	0.984
7	0.997	0.993
8	0.997	0.998
9	0.999	0.999
10	0.999	0.999
Average	0.975	0.985

Hit Accuracy

1. 16th month

Table 26 Topic Dataset 16th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	4	5	5
Ground Truth top 10	4	9	10
Ground Truth top 20	5	10	20

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	2	3
Ground Truth top 10	1	3	8
Ground Truth top 20	3	8	15

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	4	4	4
Ground Truth top 10	5	5	7
Ground Truth top 20	5	9	17

2. 17th month

Table 27 Topic Dataset 17th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	4	5	5
Ground Truth top 10	5	9	10
Ground Truth top 20	5	10	20

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	1	2
Ground Truth top 10	2	3	6
Ground Truth top 20	4	8	15

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	2	2	4
Ground Truth top 10	3	5	9
Ground Truth top 20	4	9	18

3. 18th month

Table 28 Topic Dataset 18th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	5	5	5
Ground Truth top 10	5	9	10
Ground Truth top 20	5	10	20

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	1	2	3
Ground Truth top 10	1	4	6
Ground Truth top 20	2	7	15

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	3	4	5
Ground Truth top 10	4	7	9
Ground Truth top 20	4	9	18

4. 19th month

Table 29 Topic Dataset 19th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	5	5	5
Ground Truth top 10	5	8	10
Ground Truth top 20	5	10	19

Worst hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	0	1	3
Ground Truth top 10	3	4	7
Ground Truth top 15	4	7	15

Average hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20
Ground Truth top 5	5	5	5
Ground Truth top 10	5	8	10
Ground Truth top 20	5	9	18

5. 20th month

Table 30 Topic Dataset 20th month Hit accuracy

Best hit accuracy	Prediction top 5	Prediction top 10	Prediction top 20

Ground Truth top 5	5	5	5
Ground Truth top 10	5	7	10
Ground Truth top 20	5	10	20

Worst hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	3	3	3
Ground Truth top 10	4	5	6
Ground Truth top 20	4	8	15

Average hit accuracy	**Prediction top 5**	**Prediction top 10**	**Prediction top 20**
Ground Truth top 5	1	2	5
Ground Truth top 10	3	5	10
Ground Truth top 20	4	8	18

Prediction of Specific Topics in Time series

Ground truth: red line, Prediction: blue line

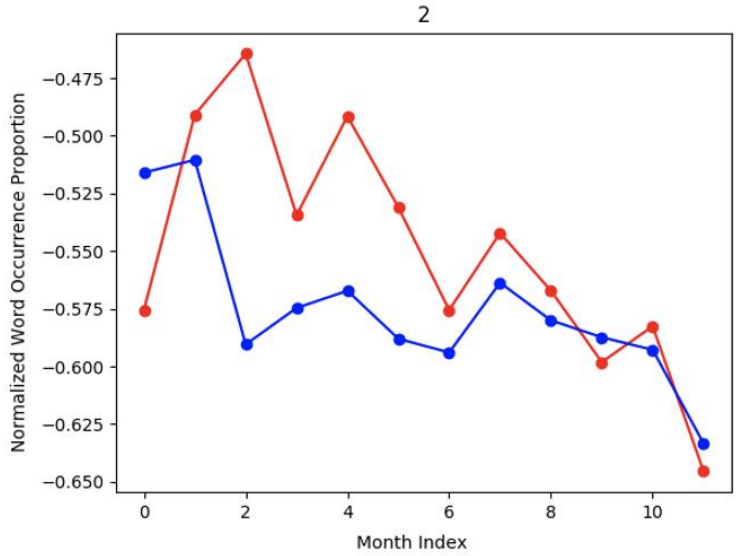


Figure 20 Topic Dataset predictions of specific topics – topic index 2

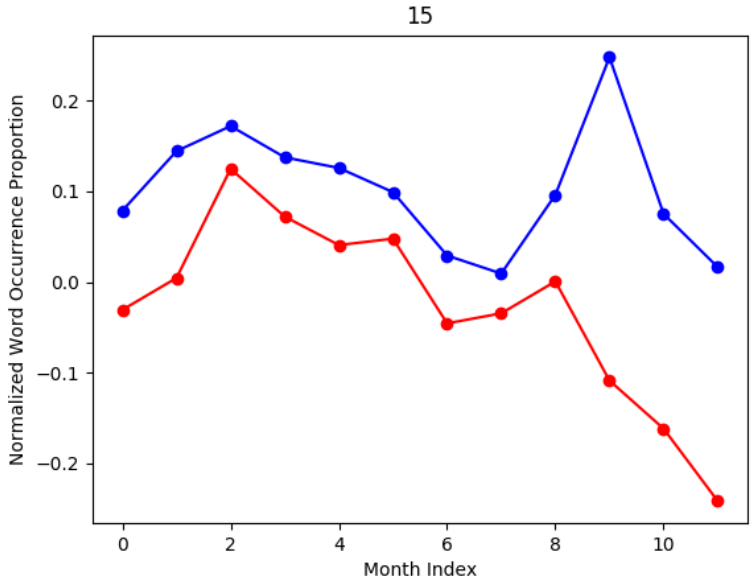


Figure 21 Topic Dataset predictions of specific topics – topic index 15

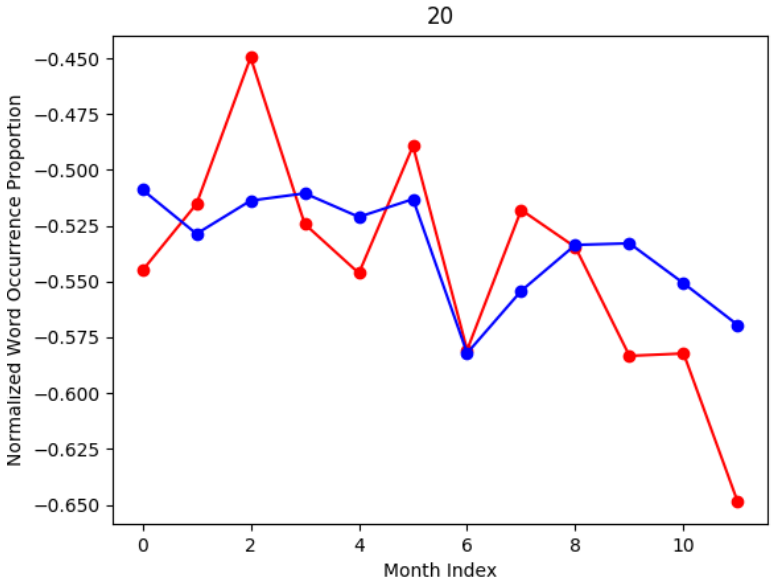


Figure 22 Topic Dataset predictions of specific topics – topic index 111

4.5 Result Analysis

There are 3 result illustrations in section 4.3 and 4.4, so the result analysis is also formed by 3 parts, as below.

For each dataset above, Pearson correlation coefficient of every testing example between prediction and its corresponding ground truth (rightmost column) is higher than the benchmark (middle column) on average. Pearson correlation coefficient of prediction is not always higher than benchmark for each example, especially when the value of benchmark is pretty high, such as 0.999, 0.997, etc. However, for those lower Pearson correlation coefficient value inside benchmark, such as 0.638, 0.731, etc., the Pearson correlation coefficient value of prediction is higher than its benchmark for most of the time. Thus, no matter how Pearson correlation coefficients vary from example to example, or from dataset to dataset, the prediction value is always more stable than the

benchmark.

As for Hit Accuracy of each dataset, although Ground Truth top 20 cannot always be predicted in Prediction top 20, Ground Truth top 5 and top 10 can be predicted in Prediction top 20 for most of the examples. Also, in most of the Best Hit Accuracy conditions, Hit Accuracy value between Ground Truth top 20 and Prediction top 20 can always become 15 to 20, which is actually pretty high. Even in worst cases, there is no circumstance that no fluctuated keyword/topic is predicted.

For each dataset, it can be shown from the figures in section 4.3 and 4.4, that prediction trend of each keyword/topic does not completely coincide with ground truth trend. However, in most cases, comparing to the ground truth trend, the prediction trend can indicate whether the specific keyword/topic becomes hot or less hot in next month, which is exactly the purpose of this project.

Chapter 5 Conclusion

This project aims to predict the trends of hot keywords and hot topics in biomedical domain with the model based on spatio-temporal graph neural networks, with large amounts of biomedical papers obtained from PubMed, the largest collection of biomedical literature.

First, titles, keywords and abstracts are extracted from each paper, together as paper contents, where hot keywords and hot topics are obtained from keyword section. Graphs of hot keywords are formed based on other 1500 add-on keywords and co-occurrence times of all hot keywords in paper contents in each month, while graphs of hot topics are formed directly from hot topics with no add-on nodes. After that, 3 datasets are acquired – Keywords(S), Keywords(L) and Topic. Then our modified model – dynamic spatio-temporal graph neural network is conducted based on hot keyword/topic frequencies (proportions) of 15 months historical data, to predict frequency of next 5 months with separate models.

The experimental results show that our modified model performs better than the benchmark evaluated on Pearson correlation coefficient in each prediction month. Also, fluctuated hot keywords/topics are able to be predicted with at least half correctness of top 20 hit accuracy of ground truth benchmark, on average, while the best top 20 hit accuracy can reach up to 20. As for the trend analysis of specific hot keywords/topics, the prediction trend of selected keywords/topics can accord with the ground truth trend mostly from the result figures.

Based on the above results, the dynamic spatio-temporal graph neural network

introduced in this project is able to collect both temporal and spatial information from historical keyword/topic frequency data. With the extracted hybrid information, the model is able to predict frequency data in next 5 months.

Chapter 6 Future Work

There are many attempts we would like to conduct in the future. First, the Pearson correlation coefficients are pretty high between ground truth of prediction month and last month in historical data of all 3 datasets, which indicates there are not much fluctuation between historical data and prediction data. Thus, we would like to try other methods to form new datasets with smaller Pearson correlation coefficient between historical data and prediction data.

Also, the dataset we currently used is papers published from year 1997 to 2016, but more recent papers, such as papers published from 2017 to 2020, would be applied in the future. Similarly, we would consider enlarging our hot keywords/topics lists, in order to generalize the model and provide with more choices.

Moreover, we would like to further modify our model. The current model conducts GCN by transforming graph data into spectral domain, which might not be compatible to larger graphs. However, as our dataset grows, the graph must grow larger. Therefore, we would like to try GCN model conducted with spatial domain mentioned in Chapter 3, which may support graph convolution on larger graphs better.

Our current predictions of next 5 months are based on separate models, which is a little redundant. In the future, we would continue to modify our model in order to predict trends of multiple months with one model.

Bibliography

1. **Home-PubMed-NCBI** [https://pubmed.ncbi.nlm.nih.gov/]
2. Han H, Giles L, Zha H, Li C, Tsioutsiouliklis K: **Two supervised learning approaches for name disambiguation in author citations**. In: *Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries, 2004: 2004*. IEEE: 296-305.
3. Dy JG, Brodley CE: **Feature selection for unsupervised learning**. *Journal of machine learning research* 2004, **5**(Aug):845-889.
4. Castelletti A, Galelli S, Restelli M, Soncini-Sessa R: **Tree-based reinforcement learning for optimal water reservoir operation**. *Water Resources Research* 2010, **46**(9).
5. Neal RM: **Connectionist learning of belief networks**. *Artificial intelligence* 1992, **56**(1):71-113.
6. Nair V, Hinton GE: **Rectified linear units improve restricted boltzmann machines**. In: *Proceedings of the 27th international conference on machine learning (ICML-10): 2010*. 807-814.
7. Graves A, Mohamed A, Hinton G: **Speech recognition with deep recurrent neural networks**. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing: 26-31 May 2013 2013*. 6645-6649.
8. Manning C, Surdeanu M, Bauer J, Finkel J, Bethard S, McClosky D: **The Stanford CoreNLP Natural Language Processing Toolkit**; 2014.
9. Ji S, Xu W, Yang M, Yu K: **3D Convolutional Neural Networks for Human**

Action Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2013, **35**(1):221-231.

10. He K, Zhang X, Ren S, Sun J: **Deep Residual Learning for Image Recognition**. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): 27-30 June 2016 2016*. 770-778.
11. Bradshaw J, Kusner MJ, Paige B, Segler MHS, Hernández-Lobato JM: **A Generative Model For Electron Paths**. *arXiv e-prints* 2018:arXiv:1805.10970.
12. Ioannidis VN, Marques AG, Giannakis GB: **Graph Neural Networks for Predicting Protein Functions**. In: *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP): 15-18 Dec. 2019 2019*. 221-225.
13. Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS: **A Comprehensive Survey on Graph Neural Networks**. *IEEE Transactions on Neural Networks and Learning Systems* 2020:1-21.
14. Veličković P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y: **Graph attention networks**. *arXiv preprint arXiv:171010903* 2017.
15. Cao S, Lu W, Xu Q: **Deep neural networks for learning graph representations**. In: *Thirtieth AAAI conference on artificial intelligence: 2016*.
16. You J, Ying R, Ren X, Hamilton W, Leskovec J: **GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models**. In: *International Conference on Machine Learning: 2018*. 5708-5717.
17. Jain A, Zamir AR, Savarese S, Saxena A: **Structural-RNN: Deep Learning on Spatio-Temporal Graphs**. In: *Proceedings of the IEEE Conference on Computer*

Vision and Pattern Recognition: 2016. 5308-5317.

18. Shi L, Zhang Y, Cheng J, Lu H: **Skeleton-Based Action Recognition With Directed Graph Neural Networks**. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): 15-20 June 2019 2019*. 7904-7913.

19. Yan S, Xiong Y, Lin D: **Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition**. *arXiv e-prints* 2018:arXiv:1801.07455.

20. Li Y, Yu R, Shahabi C, Liu Y: **Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting**. *arXiv e-prints* 2017:arXiv:1707.01926.

21. Yu B, Yin H, Zhu Z: **Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting**. *arXiv e-prints* 2017:arXiv:1709.04875.

22. Naeini MP, Taremian H, Hashemi HB: **Stock market value prediction using neural networks**. In: *2010 international conference on computer information systems and industrial management applications (CISIM): 2010*. IEEE: 132-136.

23. Fung GPC, Yu JX, Wai L: **Stock prediction: Integrating text mining approach using real-time news**. In: *2003 IEEE International Conference on Computational Intelligence for Financial Engineering, 2003 Proceedings: 20-23 March 2003 2003*. 395-402.

24. Hong T, Han I: **Integrated approach of cognitive maps and neural networks using qualitative information on the World Wide Web: the KBNMiner**. *Expert Systems* 2004, **21**(5):243-252.

25. Kianmehr K, Alhajj R: **EFFECTIVENESS OF SUPPORT VECTOR MACHINE FOR CRIME HOT-SPOTS PREDICTION**. *Applied Artificial Intelligence* 2008, **22**(5):433-458.

26. Liao R, Wang X, Li L, Qin Z: **A novel serial crime prediction model based on Bayesian learning theory**. In: *2010 International Conference on Machine Learning and Cybernetics: 11-14 July 2010 2010*. 1757-1762.

27. Corcoran JJ, Wilson ID, Ware JA: **Predicting the geo-temporal variations of crime and disorder**. *International Journal of Forecasting* 2003, **19**(4):623-634.

28. Takeuchi K, Tanaka-Taya K, Kazuyama Y, Ito YM, Hashimoto S, Fukayama M, Mori S: **Prevalence of Epstein–Barr virus in Japan: Trends and future prediction**. *Pathology International* 2006, **56**(3):112-116.

29. Liu Y, Sarabi A, Zhang J, Naghizadeh P, Karir M, Bailey M, Liu M: **Cloudy with a chance of breach: forecasting cyber security incidents**. In: *Proceedings of the 24th USENIX Conference on Security Symposium; Washington, D.C.* USENIX Association 2015: 1009–1024.

30. Chen J, Yu J, Shen Y: **Towards Topic Trend Prediction on a Topic Evolution Model with Social Connection**. In: *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology: 4-7 Dec. 2012 2012*. 153-157.

31. Wang X, Qi L, Chen C, Tang J, Jiang M: **Grey System Theory based prediction for topic trend on Internet**. *Engineering Applications of Artificial Intelligence* 2014, **29**:191-200.

32. Chen X, Xia M, Cheng J, Tang X, Zhang J: **Trend prediction of internet public**

opinion based on collaborative filtering. In: *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD): 13-15 Aug. 2016 2016*. 583-588.

33. Blei DM, Ng AY, Jordan MI, Lafferty J: **Latent dirichlet allocation**. *Journal of Machine Learning Research* 2003, **993-1022**.

34. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M *et al*: **TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems**. *arXiv e-prints* 2016:arXiv:1603.04467.

35. Gori M, Monfardini G, Scarselli F: **A new model for learning in graph domains**. In: *Proceedings 2005 IEEE International Joint Conference on Neural Networks, 2005: 31 July-4 Aug. 2005 2005*. 729-734 vol. 722.

36. Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G: **The Graph Neural Network Model**. *IEEE Transactions on Neural Networks* 2009, **20**(1):61-80.

37. Bruna J, Zaremba W, Szlam A, Lecun Y: **Spectral networks and locally connected networks on graphs**. In.: International Conference on Learning Representations (ICLR2014), CBLS; 2014.

38. Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P: **The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains**. *IEEE Signal Processing Magazine* 2013, **30**(3):83-98.

39. Niepert M, Ahmed M, Kutzkov K: **Learning convolutional neural networks for graphs**. In: *Proceedings of the 33rd International Conference on International*

Conference on Machine Learning - Volume 48; New York, NY, USA. JMLR.org 2016: 2014–2023.

40. **Laplacian matrix - Wikipedia** [https://en.wikipedia.org/wiki/Laplacian_matrix]

41. Defferrard M, Bresson X, Vandergheynst P: **Convolutional neural networks on graphs with fast localized spectral filtering**. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems; Barcelona, Spain*. Curran Associates Inc. 2016: 3844–3852.

42. Kipf TN, Welling M: **Semi-supervised classification with graph convolutional networks**. *arXiv preprint arXiv:160902907* 2016.

43. Dauphin YN, Fan A, Auli M, Grangier D: **Language modeling with gated convolutional networks**. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70: 2017*. JMLR. org: 933-941.

44. Ba JL, Kiros JR, Hinton GE: **Layer normalization**. *arXiv preprint arXiv:160706450* 2016.

45. van den Oord A, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kalchbrenner N, Senior A, Kavukcuoglu K: **WaveNet: A Generative Model for Raw Audio**. In: *9th ISCA Speech Synthesis Workshop*. 125-125.