**Wikipedia articles link prediction based on Temporal Graph Embedding**

早稲田大学大学院情報生産システム研究科

情報生産システム工学専攻 データ工学研究

44191631-2 Ma Jiaji

**Abstract**

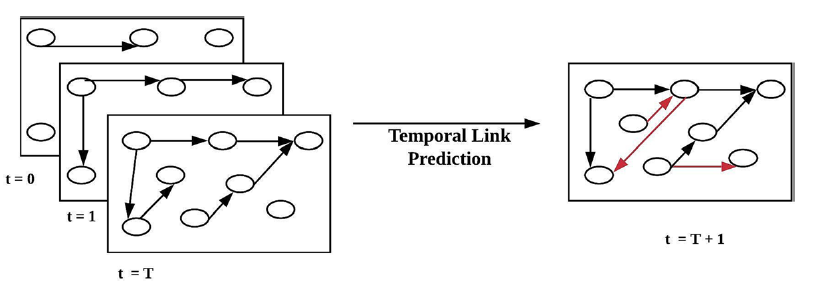
The evolutionary behavior of temporal networks has gained the attention of researchers with its ubiquitous applications in a variety of real-world scenarios. Learning evolutionary behavior of networks is directly related to link prediction problem, as the addition or removal of new links or edges over time leads to the network evolution. With the rise of large-scale temporal networks such as social networks, temporal link prediction has become an interesting field of study. In our work, we propose a temporal random walk graph embedding model and do link prediction experiments on Wikipedia links subgraph datasets. Compare with some based on random walk graph embedding model, we get higher AUC score on these datasets.

**Keywords**: Temporal Graph Embedding, Link Prediction

**Objectives of your research and backgrounds**

**Objectives**

Given a series of graph snapshot , our goal is to predict the link structure of . The node in each graph snapshot has two attributes, one is title and the other is category attribute. These two kinds of attributes are dynamic changes in each snapshot.



**Backgrounds**

Wikipedia is now one of the most popular multilingual online encyclopedias all over the world. As time goes by, a large number of articles’ interwiki links created or removed. Wikipedia has become a source of great importance for temporal text mining work.

Many researchers have proposed various methods of representation learning, trying to express complex information as vectors of fixed dimensions. The Deepwalk algorithm use the basic random walk algorithm to generate walk sequences. The Node2vec algorithm define a bias random walk strategy to control BFS or DFS of random walk, while this paper also proposes an algorithm of link prediction experiment. The LINE algorithm uses second-order proximity for directed graph, the objective function is to minimize distribution distance and use negative sampling and edge sample to optimize objective function. The SDNE algorithm use MLP neural network to train node embeddings by adjacency matrix and Laplacian matrix. The CTDNE algorithm apply for temporal graph and extend random walk into temporal random walk while also set the walk strategy. Triadic Closure algorithm aims to predict when an open triadic become a closed triadic. DynGEM algorithm use MLP neural network to train adjacency matrix, with the time goes by, gradually add one layer in the start and the end of the last neural network.

In graph embedding link prediction part, the common method is to build edge feature by binary operation and use logistic regression to predict the link.

**Problem definition mathematical formal definition**

**Temporal Graph**

We combine timestamp to ’s graph as one graph and modeling this graph as .

Let be a set of nodes as Wikipedia articles, each node , while is a list of titles with length of node , each title represent as node ’s title at time , also is a list of categories with length of node , each category represent as node ’s categories at time .

Let be a set of edges as Wikipedia article links, each link means there is a link existing from node to node at time .

**Temporal neighborhood**

The set of temporal neighbors of a node at time denoted as . , is edge ’s timestamp .

**Temporal Walk sequence**

A temporal walk sequence from to in is a sequence of titles. Randomly select a neighbor of in , and append to the title sequence list.

**Common categories**

Each node has at least one category, node and node ’s common categories at time defined as , which means common elements numbers in this two list and .

**Outline of your proposing algorithm**

The proposing model contains two parts, one is temporal graph embedding part and the other is link prediction part.

The temporal graph embedding contains three parts, initial edge selection, temporal random walk, learn node embedding.

In **initial edge selection** part, we use **unbiased strategy**, which means each edge has the same probability to be selected. We select the initial edge with the probability , which is all the links in the graph .

In **temporal random walk** part, we have two biased strategies, one is pure time bias and the other is time + common categories bias. **Pure time bias**: Given an edge , we random select one of ’s neighbor from with the probability . **Time and common categories bias**: Given an edge , we random select one of ’s neighbor from with the probability .

In **learning node embedding** part, first we generate title sequence by temporal random walk. One of title sequence such as was defined as . we generate title sequences with the maximum context window number equals to , is the nodes number of graph , defined as each node start times, is the max length of one times random walk, is the context window size. In node embedding part, we design two methods of this part, one is word embedding and the other is character n-gram embedding. **Word embedding** utilize skip-gram model to train the title sequences and get each word low-dimensional representation. **Character n-gram embedding** utilize FastText model to train the title sequences and get each n-gram character representation.

Link prediction contains four parts, node representation, train-test dataset split, feature generation, link prediction.

In **node representation** part, if we use word embedding in last procedure, the node representation is the **mean average** of each word representation of this node title, if title has new word which not in word embedding, we would ignore this word. However, if we utilize character n-gram embedding, we could also learn the word representation which was newly added.

In **train-test dataset split** part, we utilize January 1st to July 1st data to get nodes embeddings, and randomly remove 20% edges in July 20th dataset as missing links. The remaining edges defined as train positive edges and randomly select non-existing edges with the number equals to train positive edges defined as train negative edges. The missing links defined as test positive edges and randomly select non-existing edges with the number equals to test positive edges defined as test negative edges. This is one cross validation procedure and we will use **LOOCV (Leave-one-out cross-validation) method** to do 5-fold cross validation and set average AUC to be the model score.

In **feature generation** part, because our graph was directed graph, we choose **concatenation** as binary operator for learning edge features. Concatenating the embeddings is simple and preserves all information but doubles the size of the input. Also, there is a modification that we could concatenate with every two nodes **common categories one-hot encoding** as edge features.

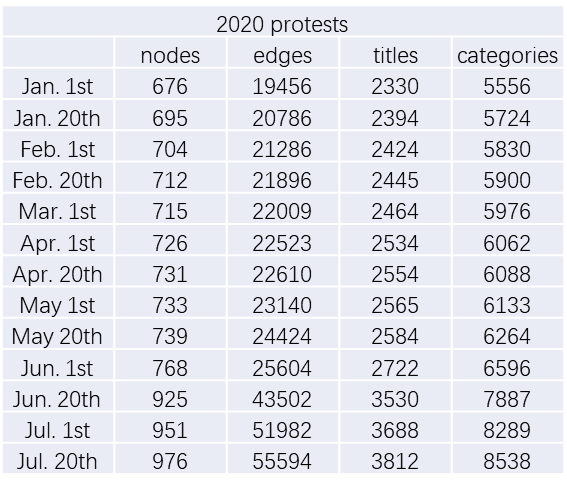
In **link prediction** part, we use **logistic regression** as a classifier to predict link condition and use AUC score as the evaluation method.

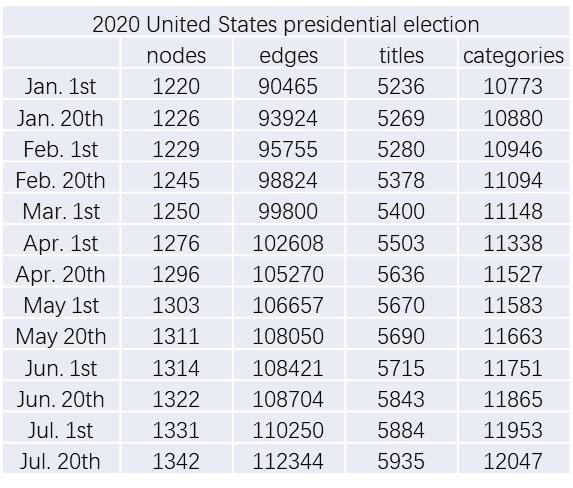
**Which point are new or different from existing methods**

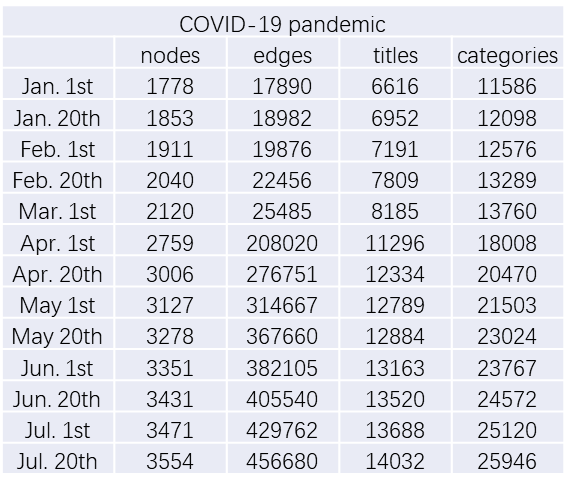
1. We use random walk to generate title sequences instead of traditional node ID sequences.
2. We define based on the number of common categories biased random walk strategy instead of traditional unbiased random walk.
3. We use character n-gram embeddings to train the title representation, which is good method for newly added word in title.
4. We both use node embeddings representation and common categories one-hot encoding as the edge features.

**Evaluation dataset**

We extracted three categories temporal link graph from whole Wikipedia dump. The timespan is from 2020 January 1st to 2020 July 20th. We collected pages number, links number, all pages title words number, all pages categories number in each timestamp.







**Current progress**

1. Learn several temporal graph embedding model and finish the coding about these baseline algorithm.
2. Extracted three subgraph datasets from whole Wikipedia dump.
3. Almost finish the code of our proposing model except character n-gram embedding and common categories one-hot encoding edge feature construction.