Peogle

Sing-Yao Wu, Hung-Hao Chen, Jheng-Huang Hong, Yun-Chih Guo
Department of Computer Science and Information Engineering, National Taiwan University

***Abstract*—There are tremendous of articles in Web forums nowadays, and many public figures have lots of positive/negative comments. We build a retrieval system which enables the users to realize the relationship between two public figures and their prestige to the public. Also, we have the relational graph to support out result so as to let the user understand fully at one glance.**

I. Introduction

Public news in Taiwan are full of contradiction and lies. It is hard to know which news is real or not, and the relations between public figures is hard to figure out. In order to help people quickly understand relationships between public figures, and help them to know popularity of each public figures, a specialized search engine with visualization aid is needed. We develop a specialized search engine with visualization of a target public figure and his or her related figures. Users can easily understand relationships between these public figures through the relational graph, and can quickly know the evaluation of each public figure through the person scores calculated from collected news in multiple media.

II. Methodology

A. Data Collection

Because we need lots of news to demonstrate our retrieval system, we collect a set of data from Taiwanese forum called ptt ourselves. The board named *Gossiping* comprises massive amounts of news posted by users everyday. We use python3.6 and its related packages such as beautifulsoup4, requests and pywordseg to crawl the data and segment the words. The format of the collected data is a json file composed of:

1) Article id: It's extracted from the URL and unique for each article.

2) Article title: The title is the summary written by reporters.

3) Article TF: It contains the counts of each word in each article.

4) Message count: We calculate the numbers of push, boo and neutral comments published by users in ptt.

5) URL: It is the link of the article.

B. Relationship Retrieval Model

We use Vector Space Model (VSM) to construct our relationship retrieval model. The input of this model is a public figure's name, and the output are *relationship edges*. Each relationship edge contains a related public figure name related to the input name, and related documents both related to these two figures.

Preprocessing we did on collected documents are removing all terms that are not Chinese names. We compute the term frequencies, inverted document frequencies from the truncated documents. We use Okapi normalization to normalize the term frequency. Input name is used as a query to search the documents related to the input name. The model use these retrieved documents to perform Rocchio feedback and expand the query. The expanded query contain only names since we truncated all the terms that is not name. We pick names with top six term frequencies in the expanded query, and use them as related public figure's name to the input name. Then we search the documents both related to input name and related figure using VSM without feedback. These documents and names are the outputs of the model.

C. Personal Score Model

For each retrieved name in our system, we calculate a score to it. Same as II-B, we use the Vector Space Model to find documents that are related to the target figure name. Then scoring each documents by machine learning technologies. The detail will be described in the following:

1) Document retrieval: The main difference between VSM in this model and the VSM in II-B is that we do not need to find the document that has relationship with other person in this model. The VSM here takes all relative documents related to the target name in count.

2) Document scoring: In order to score a document, two machine learning technologies were used: Long Short-Term Memory(LSTM) Network and fully connected neural network. Both model output a score between 0 to 100.

3) Person scoring: Many documents are relative to one person. We calculate the person's score by averaging these documents' score.

D. User Interface

In order to make our program more convenient, we design a user-friendly graphical user interface which needs only a target name string, and then it will return a related graph. The related graph contains several items, including the profiles of the target person and the most relevant people and the relational documents between them. More details are listed below.

1) person score: an int number indicates the score predicted by our Personal Score Model.

2) person image: the person's image

3) person name: the person's Chinese name

4) document title: the title of related document

5) document url: the hyperlink to the original webpage.

We implement our GUI by python tkinter, which is a very popular package for graphic design. The completed UI is Fig 1.

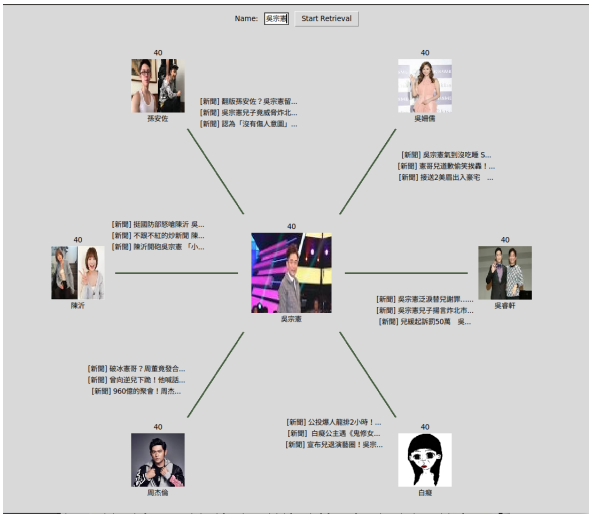

Fig. 1. The screenshot of our retrieval result.

III. Experiments

In personal scoring phase, we try two different machine learning models. Both of models are trained by 500 labeled documents and tested by 100 labeled documents. LSTM takes the document's sentences for input, and output a score between 0 to 100. Fully connected neural network takes the document's number of positive or negative reply messages for input, and also output a score between 0 to 100. We evaluate our model by calculating the average different value between predict score and ground truth.

In LSTM, the average different value is 8.7389. The predict error is less than ten percentage of 100. When we check the predict personal score, each person get the similar score, which are between 55 to 60 based on the model's parameter setting. The score 50 is the neutral score and few documents have extreme score. Although the predict error is not to large, but the predict score are very unreasonable.

In fully connected neural network, the average different value is 8.4203, which is still less than ten percentage of 100. We check the predict personal score again. Each person has more discrete predict score in this model. The fully connected neural network model's result is more reasonable.

IV. Conclusions

We develop a search engine that can visualize relationships between different public figures, and estimate the evaluation score of each figure base on comments in the internet forum. The visualization can let users get some knowledge of a figure at one glance, then from the relevant documents shown on user interface, users can easily understand why a figure's score is high or low. Currently, the function that determine if a word is a name is still weak. If we want a better performance, more advance technique will be needed. Also, the documents are only collected from news posted at single internet forum, and figure's score is also calculated from comments in same internet forum. This may be injustice to some figure that are strongly hated or liked by this internet forum. If we want to have more fair scores in the next version, we can expand the range to collect our data.