Detecting Rumors with RNN

1st Fan Yu517030910339
Shanghai Jiaotong University
fy-sky@sjtu.edu.cn

2nd Ling Qiuxuan
Shanghai Jiaotong University
2017lqx@sjtu.edu.cn

***Abstract*—Nowadays social platforms are booming, with enormous suspicious information spreading on them, and debunking rumors on social platform is a vital problem. To distinguish rumors, traditional machine learning methods are mainly depended on hand-crafted features which need a great deal of manual effort. When faced with a questionable claim, people will discuss its reliability and posting different information over time, which generates a long-distance based dependencies of evidence. Our work comes out a novel method which can learn continuous representations of microblog events to distinguishing whether such event is a rumor or not. Our model is based on the Recurrent Neural Network(RNN) which can capture the diversity of different textual information of relevant posts in the long time view. We also combines the promising self-attention method and pretrained word embedding with our model. Also, considering the fact that nowadays labeled data becomes more and more expansive to get while unlabeled data is easier to collect, we also propose a semi-supervised model to deal with such problem. Core idea of our semi-supervised model is self-training method. Experiments on real-world Weibo dataset demonstrates that (1) our supervised model outperforms all traditional machine learning method, and a simple two layers RNN network.(2) The supervised model has closely performance with the state-of-art BERT model, but training consumption is much less than BERT(Our model is trained on a 8G 1050Ti 15 minutes, while BERT is trained on a 11G 1080Ti 20minutes.(3) Our semi supervised model also beat the traditional method and 2-layer RNN, which shows a promising prospect on such method.**

***Index Terms*—RNN, Rumors Detection, Attention, Semi-supervised**

I. Introduction,Ling Qiuxuan

A rumor definition is ”a tall tale of explanations of events circulating from person to person and pertaining to an object, event, or issue in public concern.” [**?**] Rumors are always harmful as they may cause public panic and social turmoil. For example, after 2011 East Japan earthquake, a rumor comes out that the iodised salt can prevent radiation, and there will be radiation flowing from Fukushima Nuclear Power Station to China. Such rumor cause a nationwide panic in China and people rushed to purchase iodised salt blindly, and there is even someone dead because of intake too much salt in a short time. This incident of a false rumor highlights the importance of automatically distinguishing rumor in the social platform and stopping its spread.

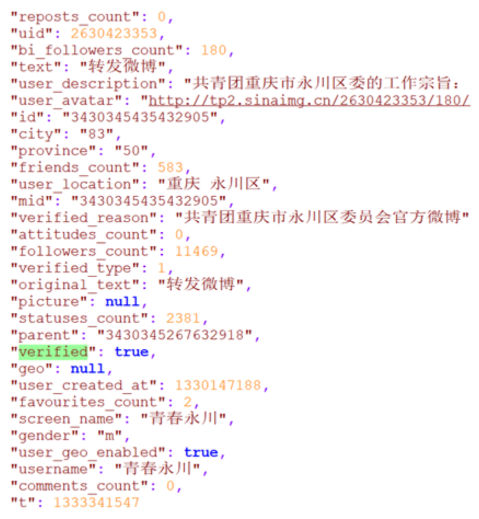
Distinguishing rumors at an early stage of spread is quite vital to minimizing their damage. To identify rumors, individuals and organizations usually involves many hand-crafted features, such as textual content, users characteristics, diffusion pattern of these posts and so on, and apply these features on traditional machine learning models, like SVM and random forest. These methods always face with two shortage:(1) first their performance is highly depended on feature engineering, which needs a great deal of human effort. (2) second, their model often have a low ability of generation, which means one special-designed and elaborate model may have a bad performance in other tasks.

On the other hand, deep recurrent neural networks have show its power in many natural language processing tasks, compared with other machine learning model. In this paper, we exploit the Bidirectional LSTM network to fully discover the hidden representation of textual content, which avoids the labor-intensive feature engineering process. By utilizing RNN, we model the textual information of an event on the social platforms as a variable-length time series. We assume people will make comments on a dubious event and offer additional information about such event. RNN will utilize the long term dependency of original text and people comments.

Considering the limits of the number of RNN units, we implement a Variable Time Series Algorithm to split the posts of an event into certain number of posters clusters. To capture the difference in diffusion pattern between rumors and facts, we add extra time features so the model learn both the temporal and textual representations from rumor posts. We also combines some promising technique, such as self-attention and pretrained word embedding, to improve our model performance. In view of the lack of labeled data and abundant unlabeled data, we also proposes a semi-supervised model based on self-training method. Extensive experiments on the rumor dataset of Weibo shows our supervised model closely performance to the state of art model BERT while computational consumption is much less than BERT, and our semi-supervised model also outperform than simple two layer RNN network.

II. Visualize, Ling Qiuxuan

The format of the data we collected is as follows. We collected the ID, parent ID, comments count, reposts count, time and facticity of each data. Then, we use the sum of the comments count and reposts count to represent the influence of the message. Also, we use the time difference of this post and the parent to represent the intimacy.



```
"reposts_count": 0,
"uid": 2630423353,
"bi_followers_count": 180,
"text": "转发微博",
"user_description": "共青团重庆市永川区委的工作宗旨:
"user_avatar": "http://tp2.sinaimg.cn/2630423353/180/
"id": "3430345435432905",
"city": "83",
"province": "50",
"friends_count": 583,
"user_location": "重庆 永川区",
"mid": "3430345435432905",
"verified_reason": "共青团重庆市永川区委员会官方微博"
"attitudes_count": 0,
"followers_count": 11469,
"verified_type": 1,
"original_text": "转发微博",
"picture": null,
"statuses_count": 2381,
"parent": "3430345267632918",
"verified": true,
"geo": null,
"user_created_at": 1330147188,
"favourites_count": 2,
"screen_name": "青春永川",
"gender": "m",
"user_geo_enabled": true,
"username": "青春永川",
"comments_count": 0,
"t": 1333341547
```

Fig. 1. Data

We take two data sets, one big and one small, and visualize them. From these figures, we can observe some characteristics of the data set.

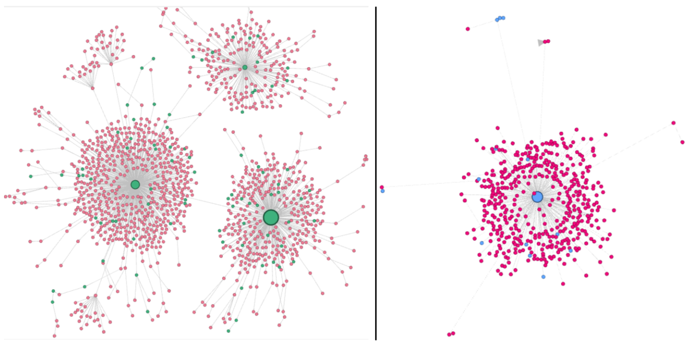


Fig. 2. visualize1

In the figures, the size of the dot indicates its influence (the bigger the dot, the greater the influence), the thickness of the line indicates the intimacy (the thicker the line, the shorter the time), and color indicates its facticity (red means the news is a rumor).

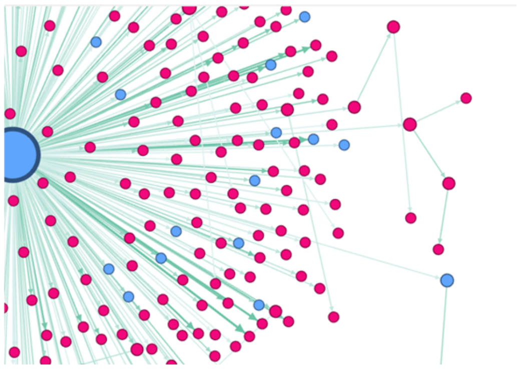


Fig. 3. visualize2

III. Model Structure, Fan Yu and Ling Qiuxuan

We explain the details of our RNN-based model for classifying microblog events as rumors or non-rumors. First we offer the definition of classifying rumor problem.

A. Problem Statement

A singal microblog post contains limited information, making it hard to classify whether it is a rumor or non-rumor. While regarding posts with the same topic as a continuous event, we can make classification on the event level, rather than the individual level. SO predicting the veracity of each post is not our interest, and we concentrate on detecing whether the event, comprised by a set of revelant posts, is rumor or not.

We define a set of given events as $E = E_i$, where each event $E_i = (m_{i,j}, t_{i,j})$ consists of relevant posts $m_{i,j}$ at timestamp $t_{i,j}$, and our task it to classify each event is a rumor or not.

B. Basic Model

To build a recurrent neural network to do classification, a very natural idea is to model each input post as an input instance, and construct a RNN with the same number as the the number of posts. However, one event may have thousands of posts, which means we need to build a RNN with thousands of units. And back propagation through such a large number of units will need high computational consumption and be likely to cause gradient explosion or gradient vanishing problem. Another idea to avoid too many RNN units is that we can split the lasting time of an event into serval equal time interval, and gather all posts in one interval together and treat it as an input instance. But such a division method may cause another problem: we all know that when an event breaks out, there will be a large number of evetns at the beginning, i.e. the first several time interval will contain plenty of posts, while the middle time interval will contain none or few posts. So such division method may cause a post unbalanced problem.

Based on such observation, we propose a Variable-length Time Series Algorithm to divide the posts, such algorithm can both solve the unbalanced posts problem and the problem of RNN units. THe input of such algorithm is all posts in one events E_i, and the expected number N of RNN units. We first initialize the length of one time interval as $l = \frac{t_{i,n_i} - t_{i,1}}{N}$. Line 4 we use divide the event lasting time into equal time intervals, line 5,6 and remove the intervals without posts. Line 7 we find the longest continuous time span interval. Line 8 if such interval is shorter than N and current interval is longer than the last interval, it means we can continue spliting the event with shorter time interval, so line 9 we shorten the length of time interval by 50%. Otherwise we just return the result we split. No matter how long an event lasts, the number of our final output time interval is about N, and each time interval has the same length.

```
Input  : Relevant posts of $E_i = \{(m_{i,j}, t_{i,j})\}_{j=1}^{n_i}$,
         Reference length of RNN $N$
Output: Time intervals $I = \{I_1, I_2, \ldots\}$
   /* Initialization                                    */
1  $L(i) = t_{i,n_i} - t_{i,1}$;  $\ell = \frac{L(i)}{N}$;  $k = 0$;
2  while true do
3      $k$ ++;
4      $U_k \leftarrow Equipartition(L(i), \ell)$;
5      $U_0 \leftarrow$ {empty intervals} $\subseteq U_k$;
6      $U'_k \leftarrow U_k - U_0$;
7      Find $\bar{U}_k \subseteq U'_k$ such that $\bar{U}_k$ contains continuous
         intervals that cover the longest time span;
8      if $|\bar{U}_k| < N$ && $|\bar{U}_k| > |\bar{U}_{k-1}|$ then
           /* Shorten the intervals                     */
9          $\ell = 0.5 \cdot \ell$;
10     else
           /* Generate output                           */
11         $I = \{I_o \in \bar{U}_k | I_1, \ldots, I_{|\bar{U}_k|}\}$;
12         return $I$;
13     end
14 end
15 return $I$;
```

Algorithm 1: Algorithm for constructing variable-length time series given the set of relevant posts of an event and the reference length of RNN

After the segmentation of event posts, we can build a basic RNN model using the output of Variable-length Time Series algorithm as input. Note we can replace the RNN unit with LSTM unit or GRU unit.

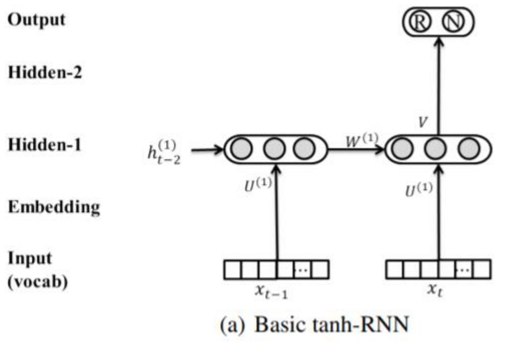


Fig. 4. Basic Model

C. Diffusion Pattern

As an old saying goes ”Truth goes on crutches, while rumor has wings”. There exists some difference between diffusion pattern of rumor and truth Can we capture the feature of diffusion pattern and combine it with textual feature to help improve the performance of our classifier? Take the result of our visualization as an example, if an event is transmitted many times in a short period, the cluster will be small and dense, the event is diffused quickly and gain lots attention,

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} Z_j^{(i)}$$

$$\sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}(Z_j^{(i)} - \mu_j)^2$$

$$\hat{Z}_j = \frac{Z_j - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}$$

Fig. 5. Different Event Diffusion

if an event is transmitted slowly in a long period, the cluster will be large and sparse, the event is diffused slowly and gains less attention. So we add an additional time feature as the description of the diffusion pattern and concate it with the textual feature. We calculate the mean and standard deviation of all posts' time in a single interval, and utilize the idea of batch normalization to normalize all posts' time in a single interval, after normalization, the distribution of posts' time can be a reflection of the posts cluster's density, i.e. the diffusion pattern.

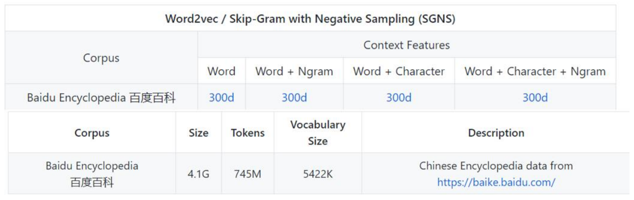
$$\mu_i = \frac{1}{m}\sum_{j=1}^{m} t_{i,j} \tag{1}$$

$$\sigma_i^2 = \frac{1}{m}\sum_{j=1}^{m}(t_{i,j} - \mu_i)^2 \tag{2}$$

$$\hat{t_{i,j}} = \frac{t_{i,j} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}} \tag{3}$$

D. Word Embedding and Self Attention

As our corpus is small, we think it may be not enough to train a fruitful word embedding. So we consider the idea of transfer learing and use a well pretrained word embedding to replace our own embedding and finetune on other layers. The word embedding we use is trained on the 4.3G Baidu Encyclopedia with 745M tokens and 5422K vocabulary, it's a word2vec word embedding with Character and Ngram.



Word2vec / Skip-Gram with Negative Sampling (SGNS)

Corpus	Context Features			
	Word	Word + Ngram	Word + Character	Word + Character + Ngram
Baidu Encyclopedia 百度百科	300d	300d	300d	300d

Corpus	Size	Tokens	Vocabulary Size	Description
Baidu Encyclopedia 百度百科	4.1G	745M	5422K	Chinese Encyclopedia data from https://baike.baidu.com/

Fig. 6. Word Embedding

Nowadays attention mechanism is widely used in natural language processing, it can alleviate the problem that the model can't build long distance dependency by using different weights in processing different RNN units. In this paper, we don't use the traditional attention mechanism, but we use the promising self attention mechanism which has a better ability of capturing internal information. Self attention is first proposed by Google 2017 in the paper $Allyouneedisattention$. Unlike traditional attention mechanism, the query Q, key K, value V in self attention is from the same input. To calculate self attention, we dot input X with parameter W_Q, W_K, W_V and get the output Q, K, V, then calculate the scale dot of Q and K, and pass the dot result through a softmax operation, and finally dot V to get the output attention distribution.

$$Q = W_Q Q \tag{4}$$

$$K = W_K K \tag{5}$$

$$V = W_V V \tag{6}$$

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{7}$$

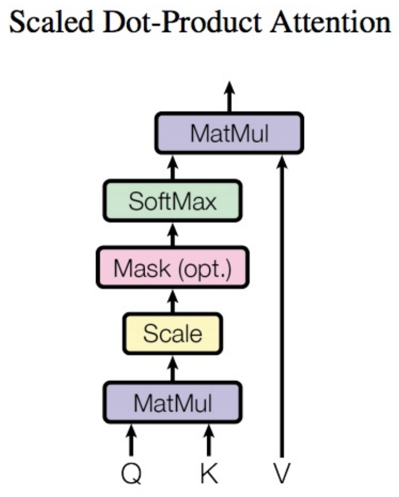


Fig. 7. Self Attention

E. Supervised Model

After adding diffusion feature, pretrained word embedding and self attention mechanism, we can build a semi supervised model, which consists of a embedding layer, transferring input instance and their time feature into textual feature and diffusion pattern feature, a bi-direction LSTM layer, a self-attention layer, calculating attention distribution for different position, then a bi-direction LSTM layer, a dropout layer to avoid over fitting and finally a softmax layer to ouput prediction results.

F. Semi Supervised Learning

Nowadays, labeled data becomes more and more hard to get, as manual hunman labeling process is more and more expensive. While at the same time, we have easier access to get enormous unlabeled data from society. Faced with such a dilemma, we propose a semi-supervised learning model

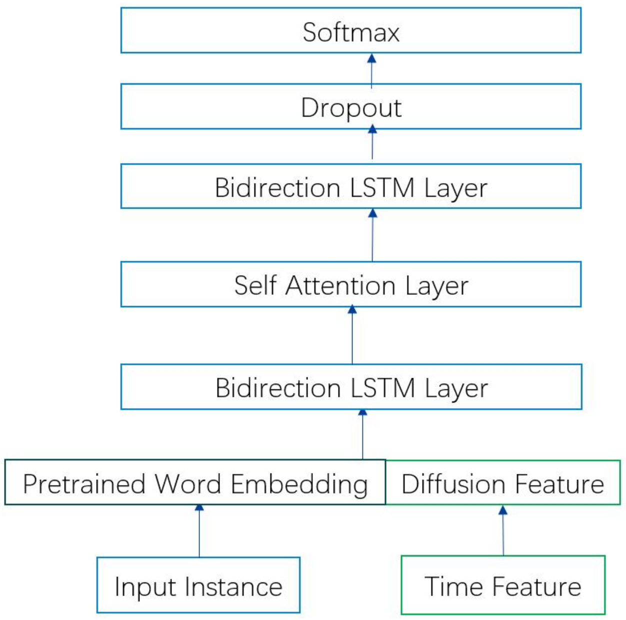


Fig. 8. Model Structure

to make fully utilization of unlabeled data and expand our model's capability. The core idea of our semi-supervised learning model is "self training". First line 4,we use labeled data to train a classifier.Then line 5 we use this classifier to make prediction of unlabeled data, line 6 we pick the results with high confidence and label these results with the predicted labels, and line 7 remove them, finally line 8 we combine these data with orginal data.

Algorithm Self-training.

```
1: Initialize:
2: Given $(X_{train}, y_{train}) = (X_l, y_l)$
3: while stopping criteria not met do
4:     Train classifier $C_{int}$ from $(X_{train}, y_{train})$
5:     Use $C_{int}$ to predict class label $y_u$ of $X_u$
6:     Select confidence sample $(X_{conf}, y_{conf})$; $(X_{conf}, y_{conf}) \in (X_u, y_u)$
7:     Remove selected unlabeled data $X_u \leftarrow X_u - X_{conf}$
8:     Combine newly labeled data $(X_{train}, y_{train}) \leftarrow (X_l, y_l) \cup (X_{conf}, y_{conf})$
9: end while
```

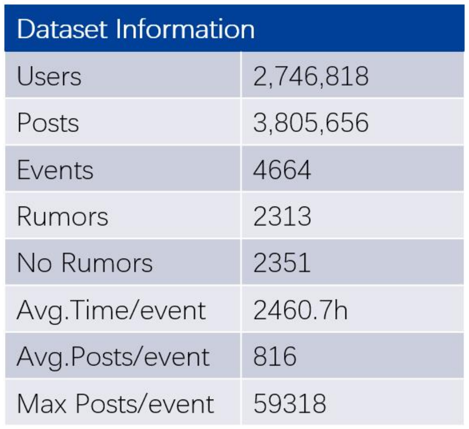
IV. Experiments, Fan Yu

A. Dataset

Our dataset is from the the Sina community management center, which reports misinformation. Our dataset contains 3.8 million posts and 4664 events, with 2313 rumors and 2351 non-rumors. And we randomly split the dataset by 7:3, i.e. we have 3264 train examples and 1400 test examples. The details of the dataset is followed:

B. Baseline Design

To make fully comparision, we implement several tradional machine learning method, including logistics regression, random forest, GBDT and KNN, a basic 2 layer RNN, and



Dataset Information	
Users	2,746,818
Posts	3,805,656
Events	4664
Rumors	2313
No Rumors	2351
Avg.Time/event	2460.7h
Avg.Posts/event	816
Max Posts/event	59318

Fig. 9. Dataset Information

the state-of-art model in many natural language processing, BERT(Bidirectional Encoder Representations from Transformers). The general framwork of BERT and transformer is showed in

Our framework is Keras 2.24 and Tensorflow 1.13.1. The

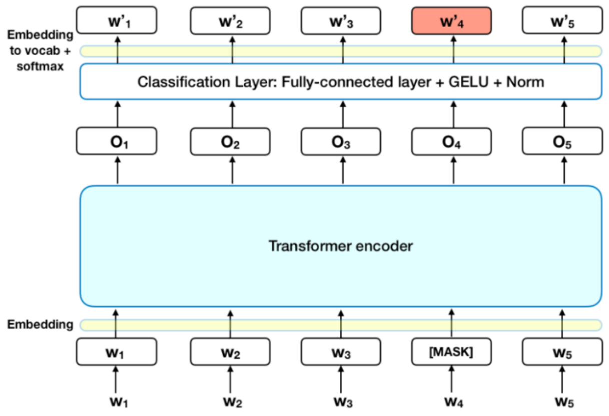


Fig. 10. BERT Structure

results are followed:

From the results, we can find that our supervised model has close performance to the state of art model BERT, and considering the computational consumption between BERT(trained on 1080ti for 4 hours) and our model(1050ti for 3 hours), our model is much cheaper and easier to train. Also, our semi-supervised model still outperforms the traditional RNN, which means we have boarden our model's generalization ability while not hurt its performance. Also, both supervised model and semi-supervised model have much better performance than traditional machine learning method.

V. Conclusion and Future Work, Fan Yu

Our contribution in this project can be summarized as follows:

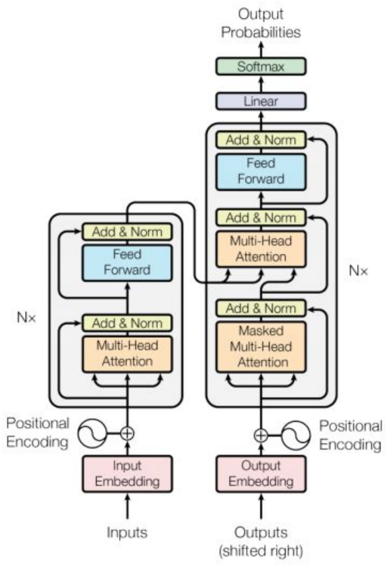
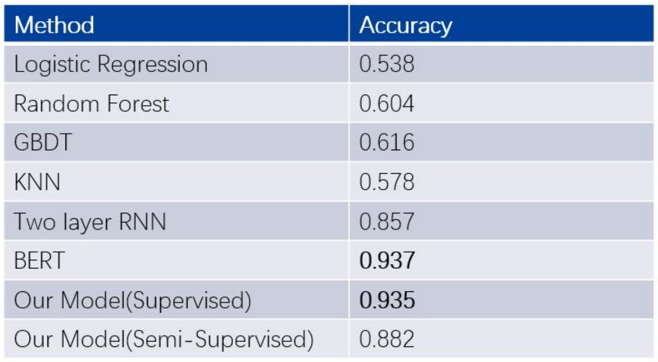


Fig. 11. Transformer Structure



Method	Accuracy
Logistic Regression	0.538
Random Forest	0.604
GBDT	0.616
KNN	0.578
Two layer RNN	0.857
BERT	0.937
Our Model(Supervised)	0.935
Our Model(Semi-Supervised)	0.882

Fig. 12. Experiment Result

1) We implement the Variable Time Series Algorithm to divide the posts of single event into certain number same-length time slots.
2) We consider the different diffusion pattern of rumor and non-rumor events and add additional time features with the text features to help caputure such features.
3) We utilize self-attention mechanism and pretrained word embedding to help improve our model performance.
4) We propose a semi-supervised model, based on the self-training algorithm, to alleviate the problem of the lack of labeled data. And both the supervised model and unsupervised model have good performance.

As for the future, we think our model can be improved in the following two directions:

1) It's not very accurate to use normalized time features to learn the representations of diffusion pattern. And nowadyas has already witnessed a boost in graph convolutional neural network(GCNN), which shows a strong ability to capture information in graph. And combining

GCNN method with our model may achieve better performance in rumor diffusion patter capture.

2) Except self training, there are many other semi-supervised learning method, such as co-training, tri-training and so on. And we can try these semi supervised learning method to further explore our semi-supervised model.

Workload: Fanyu: model design, baseline design, 60%
Ling Qiuxuan: visualization, Basic RNN design, baseline design, 40%

References

[1] Jing Ma et al. "Detecting rumors from microblogs with recurrent neural networks." In: Ijcai. 2016, pp. 3818– 3824

[2] Ashish Vaswani et al. "Attention is all you need". In: Advances in neural information processing systems. 2017, pp. 5998–6008.

[3] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: arXiv preprint arXiv:1810.04805 (2018).

[4] C Lin et al. "Self-training improves recurrent neural networks performance for temporal relation extraction" In: Association for Computational Linguistics pages 165–176