COMP90042 Project Report

1126832

University of Melbourne

1. Introduction

The concept of rumour has a long history, and it involves a form of a statement whose veracity is not quickly or ever confirmed. Rumours have the potential to spread quickly through social media, and bring about significant economical and social impact. In this report, we build a classifier to classify the rumours in the tweets based on the text and user information of source of replies. Furthermore, some analysis is performed with using this classifier to understand the nature COVID-19 rumours and the difference between the rumours and non-rumours.

2. Dataset

We build the rumour classifier on tweet dataset. Each sample in the tweet dataset is an event: a list of tweets from the event where the first tweet is the source and the rest are reply tweets. Each tweet contains the text and some other information such as user information, creating date, whom this tweet replies to, etc. The tweet dataset contains 4641 training samples, 580 validation dev samples and 581 test samples.

We analysis another tweet dataset about COVID-19 rumours.

3. Methods

3.1 Data preprocessing

The models in my experiments include into three kinds: machine learning models, LSTM network and Bert related network. The data preprocessing of them are different, and I will describe both of them in the following. Firstly, for each event sample, it is divided into two parts: source text and reply texts sorted by creating date or by number of followers of user. Then remove urls and @ from both sources and replies because both of them contain many random characters which are meaningless for NLP task.

For machine learning models, the texts are tokenized into tokens by a pretrained tokenizer, then remove the tokens contain punctuations. All the tokens are turned to be lowercase and stopwords are removed. At last, lemmatize all the tokens so that these tokens are suitable for TF-IDF transformation.

Instead of bag-of-words model, LSTM considers the order of tokens in the sequence, so the tokenizer is fit on the training set. Use the tokenizer to convert each token into token_id. In the end, restrict the token_id sequences to a specific length of 25.

For Bert model, the tokenizer from bert-base-uncased can produce suitable tokens for bert model. The bert-base-uncased tokenizer was pretrained on BookCorpus dataset which consists of 11038 unpublished books and English Wikipedia using a masked language modeling objective. Since the tokenizer is "uncased", we don't need to do the lowercase operations on the inputs. Unlike rule-based model including classifier based on TF-IDF values, Bert doesn't benefit from some processing including stopwords removal and lemmatization especially some stopwords may change the meaning of sentence such as not, no, etc. The source sentence and replies sentence are both restricted to specific length respectively by padding [PAD] or cutting, and source sentence and replies sentence of them constitute an input sequence. [CLS] is put at every input sequence, [SEP] is put at the end of each sentence.

3.2 Experiment setup

We test and compare the performance of 2 machine learning models and two neural network models: Logistic Regression, Support Vector Machine Classification (SVC), Long short-term Memory (LSTM) and Pre-training of Deep Bidirectional Transformers (Bert). Logistic Regression is a common classification model, so here it is considered as baseline; Inspired by Kahyun Choi who found SVM performs well for text features [1], SVM is considered; LSTM can "remember" the precious knowledge to capture the dependency between tokens; Instead of surface representation from LSTM, Bert can capture the contextual representation, which is the state-of-art method for many natural language processing tasks.

Machine learning models: After the data preprocessing, the features consist of TF-IDF values of every token. We try two strategies to train logistic regression and SVC: training the machine learning model 1. only with sources and 2. with both sources and replies. Grid search is used to tune the hyperparameters by the cross-validation score with 5 folds.

LSTM : We build a network with one LSTM layer, the structure is shown in right figure. The first layer is used to embed each token with word embeddings of 128 length. The

word embedding is fed to the LSTM layer. Dropout layer is added after the fully-connected layer to avoid overfitting.

Bert : Bert is pre-trained on large corpus to learn knowledge, and can use them in our rumour classification task. It uses self-attention networks to capture long dependencies between words, so that we can feed more replies into the network compared to LSTM. Furthermore, Bert can capture the contextual representation instead surface representations from LSTMs. Each input sequence contains the one source tweet and the combination of all replies from an event. We feed the input sequence to Bert, and take the contextualized embedding of [CLS] produced by Bert layer and pass it to a linear layer to get the output. The output is a single scalar value denoting the probability of positive class after sigmoid operation. We experiment on different fixed length of source and replies, different epoch numbers, different network structure, etc. We will experiment on how many tokens from sources and replies should be took as the input sequence, and compare the performance of different optimizers on this task.

```
Model: "lstm"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_11 (Embedding)     (None, 25, 128)           1443072
_________________________________________________________________
lstm_10 (LSTM)               (None, 128)               131584
_________________________________________________________________
FC1 (Dense)                  (None, 128)               16512
_________________________________________________________________
dropout_6 (Dropout)          (None, 128)               0
_________________________________________________________________
FC2 (Dense)                  (None, 1)                 129
=================================================================
```

4. Results and Evaluation

As mentioned in Section 3, the result of experiment is shown is the below table 1. We tune the hyperparameters for each model, and list the one with the highest f1 score on test set.

	logistic regression	SVC	Bert	LSTM
precision	0.8690	0.8105	0.781	0.7526
recall	0.6738	0.8191	0.872	0.7766
f1	0.7590	0.8148	0.824	0.7644

Table 1: test results of 3 models

From the table, logistic regression (LR) model gives a quite low result. One of the possible reasons is that it is not effective for non-linear data, also LR takes the TF-IDF features which cannot capture the semantics of text. However, the fact that SVC with radial basis function (RBF) kernel got the second highest score among four models although it is fed by TF-IDF features. The main advantage of SVR is that it uses RBF kernel to transform the non-linear data into higher-dimension space where the data become linear. Furthermore, SVC can find the hyperplane which maximum the margin from data points. The above two reasons make the SVC perform better greatly than LR model. In addition, we find machine learning models (SVC and LR) perform worse on TF-IDF features of both sources and replies, it may because many replies are consistent the previous text in semantics but have little meanings as the single sentence, but TF-IDF can't capture the semantics well. The f1 score of machine learning models on different texts is shown in table 2 below.

	logistic regression	SVC
only sources	0.7590	0.8148
sources and replies	0.6667	0.7439

Table 2: machine learning models on different text

The LSTM model doesn't perform well on the task, which may due to inappropriate parameter and hyperparameter tuning. However, LSTM is unable to capture long range dependencies and may have gradient vanishing/exploding, which may restrict the upper bound the LSTM model.

The Bert model performs the best among all models, which achieve 0.824 f1 score on test set. We make many experiments for the Bert model, including comparing the performance different optimizers(BertAdam, Adam, AdamW); adjusting the hyperparameters(learning rate, maximum length of source tokens and replies tokens, decay rate, etc); putting the dropout layer between Bert and linear layer. Due to the word limitation, we don't describe in details, but the implementation is shown in code. This is the setting for the best bert model: 1. The AdamW optimizer [2] is used to optimize the cost function with weight decay; 2. 30 tokens from source and 100 tokens from replies are took as the input sequence. 3.the replies are sorted according to creating date and number of followers. Since Bert can capture longer dependency than LSTM, I take 30 tokens from source and 100 tokens from replies after tuning.

5. Discussion and Future Work

We try several kinds of features and several kinds of models for this rumour classification task. Some classic machine learning models (such as SVC) can perform normal on text with TF-IDF features on this task, however there is little space to improve for them. The Bert related network I build in this report is not a complicated one, but it performs the best among other models, so it's obvious that the contextual representation produced by Bert is powerful for this kind of General Language Understanding Evaluation (GLUE) task. The Bert network has a great number of parameters and hyperparameters, which is a challenge to tune the network, such as the choice of optimizer, initialization strategy, network structure, etc. There are also some other state-of-art techniques useful for text classification besides pre-training task, such as capsule neural network that deals with

information loss of pooling and mixture model that combines attention, RNN, CNN. [2]

6. Analysis of COVID-19 tweets

In this section, we use our best model(i.e. bert model) to perform some analyses on a COVID-19 tweets dataset to understand the how rumours and non-rumours differ from each other.

6.1 similarities and differences between hashtags of rumours and non-rumours

Hashtag as a popular function in Twitter categorizes Tweets by keyword. In the first step, we focus on the hashtags. We look at the popular hashtags for rumours and non-rumours respectively and analyze the overlap and difference between them.

We obtain 24591 hashtags from non-rumours and 2257 hashtags from rumours, then take the most frequent 100 hashtags in COVID-19 rumours and COVID-19 non-rumours respectively. One interesting finding is that there are 62 overlapping hashtags between them, which doesn't show the significant difference between hashtag types rumours and non-rumours. It is important to bear in mind that in this task, the rumours and non-rumours are both related to COVID-19, so it's common they share some hashtags heavily connected to COVID-19 such as "covid19", "coronaviruspandemic", "wuhanvirus
", "stayhome" (shown in Figure 3). Even for some rumours, they can also have hashtags but pass some fake information about the COVID19-related hashtag. The most frequentist overlapping hashtags are shown in Figure 3. The second and third column represents the frequency of corresponding hashtag in rumours and non-rumours. What obviously stands in this figure is that the frequency of hashtag of non-rumours is commonly higher than that of rumours. For example, the most two frequentist hashtags ("covid" and "coronavirus") have frequencies of 1.99 and 1.11 in non-rumours, but only 1.07 and 0.65 in rumours. The average frequency of top 100 hashtags of rumours is 0.030, while 0.054 for non-rumours. The result indicates that rumours have a wider range of hashtags and the distribution of hashtags is less central than for non-rumours. A possible explanation for this might be that may non-rumours are published by some official organization such as news agency and government, these kinds of organizations need to be standard to maintain good image in the front of the public so they tend to use some standard terms as the hashtags, while many rumours are usually published by individuals or some unofficial organizations and may be more casual than non-rumours. Another possible explanation is that non-rumours describe the fact which is always unique, but rumours can distort the fact in many aspects.

Figure 4 show top 10 most frequentist hashtags in rumours but not for non-rumours, and Figure 5 show top 10 most frequentist hashtags in non-rumours but not for rumours. For the 10 hashtags of non-rumours, hashtags are directly related with COVID-19 except some hashtags about Donald Trump, and Donald Trump is involved in many news about COVID-19. For hashtags of rumours, the types of hashtags are wider. For example, "breaking" is a general hashtag, which means people can't infer COVID-19 from "breaking" hashtag; "georgefloyd" hashtag is about an event between policemen and a driver, indirectly related to COVID-19, but has a high frequency in COVID-19 rumours. These two figures indicate that non-rumours are stricter and more related to the COVID-19 than rumours in this dataset.

hashtag	rumour_freq	nonrumour_freq
covid19	1.076498	1.994256
coronavirus	0.655363	1.113033
trump	0.068612	0.159543
coronaviruspandemic	0.046530	0.096912
wuhanvirus	0.046530	0.090241
covid—19	0.046530	0.064855
covid_19	0.044164	0.074305
covid	0.035489	0.065473
trumpvirus	0.029180	0.091538
china	0.027603	0.047498
trumpliespeopledie	0.025237	0.042248

Figure 3: common hashtags both in rumours and non-rumours; The second and third column represents the frequency of corresponding hashtag in rumours and non-rumours.

hashtag	nonrumour_freq
socialdistancing	0.019951
trumpisanidiot	0.018653
trumpisaloser	0.018592
contest	0.018221
stayathome	0.016677
healthcareheroes	0.016245
kag	0.015689
trumpfailedamerica	0.015503
dumptrump2020	0.015503
trumpisalaughingstock	0.014577
votebluetoendthisnightmare	0.013774

Figure 4: hashtags only in rumours; the second column represents the frequency of corresponding hashtag in non-rumours

hashtag	rumour_freq
breaking	0.029180
georgefloyd	0.022082
wuhan	0.014196
trumpandemic	0.012618
covid19ph	0.009464
riots2020	0.009464
firetrump	0.009464
dominiccummings	0.008675
india	0.007886
hoax	0.007886
fauci	0.007098

Figure 5: hashtags only in non-rumours; the second column represents the frequency of corresponding hashtag in non-rumours

6.2 **characteristics of rumour-creating users and nonrumour-creating users**

Twitter API provides some information of tweet-creating users which can be useful to compare the difference of rumour-creating users and nonrumour-creating users, thus have a general understanding of characteristics of rumour-creating users.

Firstly, we divide the COVID-19 into rumours and non-rumours by my best model of task 1. Then from each source, we extract 6 types of user information: The number of followers this account currently has, the number of users this account is following, the number of public lists that this user is a member of, the number of Tweets (including retweets) issued by the user, the number of Tweets this user has liked in the account's lifetime, and whether the user has a verified account. Then compute the mean and median of each except that verified attribute just has percentage because it's a binary attribute.

As shown in Figure 6, for each statistic, mean is significantly greater than median because there are some users who have a large number of followers, friends, etc. Hence, we focus on both median and mean for this task. On "followers count", "friends count", "list count" attributes, they share the similar pattern: the median among rumours-creating users is similar to that among nonrumours-creating users, but mean among nonrumours-creating users is greater than that among rumours-creating users with statistically significance. This phenomenon illustrates the fact that there are more users who have great followers and posts publishing non-rumours than those publishing rumours. A possible explanation for this might be that some official organizations such as BBC are easy to have a lot of followers, but those who post rumours are hard to attract people. Furthermore, some rumours may have serious impact, if a rumour a seen by many people, Twitter may block the account to preserve the good media environment. Unlike "followers count", "friends count", "list count", "statuses count" have different patterns. The mean of "statuses count" among rumours-creating users is around 30% greater than that among nonrumour-creating users, the median is even 45% greater. The statuses count represents the number of Tweets (including retweets) issued by the user, this result may be explained by the fact that many rumours-creating users tend to be active and make many tweets(or retweet), maybe because some of them want to spread the rumours. One interesting finding is that the verified percentage of rumour-creating users is 2% higher than that of nonrumours-creating users, which contradicts to our intuition that those who post rumours are unwilling to verify the account. However, we need to bear in mind that many accounts are unverified because their users don't use twitter often, while those who publish rumours by twitter is more likely to use twitter often. In a word, we don't control the activeness at the same level when performing the comparison, and activeness is highly related to whether to "verified".

		followers count	friends count	list count	statues count	favourite count	verified
rumours-creating users	mean	4,136,863	6,966	14,546	144,600	24,091	76%
	median	408,805	1,040	2,162	62813	2829	
nonrumours-creating users	mean	5,447,267	7,935	15,522	113,796	24,288	74%
	median	387,772	1,106	2,202	39039	3164	

Table 6: some Statistics of rumour-creating users and nonrumour-creating users

6.2 Summary of Analsis

We analyze the COVID-19 rumours from twitters from two aspects: hashtags and user information. All the conclusions are based on my classifier, but my classifier has 0.8241 f1 score on colab competition with 0.781 precision and 0.872 recall on test set. The high recall and low precision indicate that my model may predict some non-rumours to be rumours, which may influence the analysis in the section 6. Firstly, we find that rumours and non-rumours share most of the hashtags, but rumours tend to have wider variety of hashtags and the distribution of hashtags is less decentralized than non-rumours. Secondly, there are many people who has many followers publish non-rumours tweets, and rumour-creating users tend to be more active than nonrumour-creating users.

Reference

[1] K. Choi, "Music Subject Classification".

[2] huggingface, "https://huggingface.co/transformers/_modules/transformers/optimization.html#AdamW," huggingface, [Online]. Available: https://huggingface.co/transformers/main_classes/optimizer_schedules.html?highlight=adam#adamw-pytorch.

[3] N. K. Shervin Minaee, "Deep Learning Based Text Classification: A Comprehensive Review," 4 1 2021.

[4] huggingface, "huggingface.co," [Online]. Available: https://huggingface.co/transformers/model_doc/bert.html#flaxbertforpretraining.