Fake News Stance Detection

Dilip Molugu
George Mason University
Fairfax, VA, USA
dmolugu@gmu.edu

Vijayasaradhi Muthavarapu
George Mason University
Fairfax, VA, USA
vmuthava@gmu.edu

Nikhil Reddy Pathuri
George Mason University
Fairfax, VA, USA
npathuri@gmu.edu

Abstract

Using machine learning techniques in text classification has been really useful nowadays. Using various machine learning techniques like CNN, SVM have been proven to work well in in classifying the text with the word embeddings. Therefore, we have implemented to test the relation between the headline and the body to detect if it is a fake news or not by finding the stance of the relation between the headline and the body. Our main goal is to suggest a model which will classify the kind of relation between the headline and the body of news. This kind of classifying will somewhat what help us to fight against the fake news.

1 Credits

The dataset we used for this problem is taken from the Fake News Challenge website. The website held the competition last year and awarded a cash price of $10,000 to the winners. This competition is indented to create and encourage new ideas to find new solutions for fighting fake news problem.

2 Introduction

In recent years there is an increasing problem with the Fake news and the problems caused by them. Fake news is a type of yellow journalism or propaganda that consists of deliberate disinformation or hoaxes spread via traditional print and broadcast news media or online social media. Fake news, in this age of the internet has evolved in such a way that it has become hard even for experts to detect
the fake news. Fake news was used in many cases as a political tool to manipulate the views of the people. Due to this manipulative use of the fake news detection needs to be given a lot of importance and cannot be ignored at any cost. though this might not look like affecting anyone in some situations, there is a bigger problem which goes undetected. In many countries this w is being used by government to promote their ideas and by to manipulate the mindset of their citizens.

Facebook in recent years has been playing a major role in spreading the fake news. It has been widely used by organizations and individuals as a media to spread the news. Facebook is also finding a way to completely stop the spreading of the fake news. Using the machine learning techniques makes it easier to identify the fake news stances like agreed, disagreed, discussions and unrelated when trained with proper training data and when used with proper classifiers.

This process of classifying the news will also help us understand which sources are providing fake news and will allows us to identify the sources producing the fake news and this might in future can be used as new feature to identify the fake news. The authors and the frequency of such posts on websites and many other features can also be added in future increase the accuracy of the predictions. This kind of classifying will allow us to automate the process in the future.

In this paper we are going to evaluate using CNN to train the classifiers to identify the relationship between the body and the headline of the news. The data will be trained on the data which had labels like agree, disagree, discuss and unrelated.

3 Data Component

We acquired the data required for this project from a fake news challenge competition (Source- https://github.com/uclmr/fakenewschallenge). The data is well prepared and contains training and testing splits. They provided the key for the test data. Although the data is clean, we need to pre-process it according to our models. Typically, our dataset contains Headlines Bodies and related stances.

1	Headline	Body_ID	Stance
2	Police find mass graves with at lea	712	Unrelated
3	Hundreds of Palestinians flee flooc	158	Agree
4	Christian Bale passes on role of Ste	137	Unrelated
5	HBO and Apple in Talks for $15/Mc	1034	Unrelated
6	Spider burrowed through tourist's	1923	Disagree
7	'Nasa Confirms Earth Will Experier	154	Agree
8	Accused Boston Marathon Bombe	962	Unrelated
9	Identity of ISIS terrorist known as '	2033	Unrelated
10	Banksy 'Arrested & Real Identity Ri	1739	Agree

Training_data

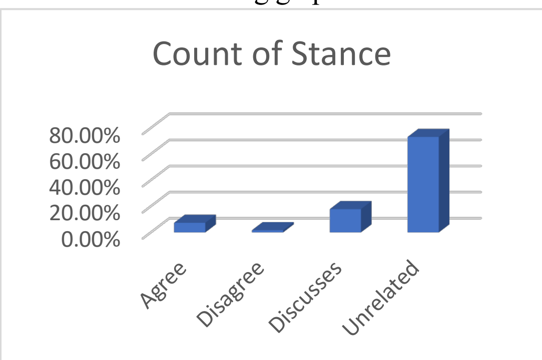
1	Headline	Body_ID
2	Ferguson riots: Pregnant wom	2008
3	Crazy Conservatives Are Sure	1550
4	A Russian Guy Says His Justin I	2
5	Zombie Cat: Buried Kitty Belie	1793
6	Argentina's President Adopts I	37
7	Next-generation Apple iPhone	2353
8	Saudi national airline may intr	192
9	'Zombie Cat' Claws Way Out (	2482
10	ISIS might be harvesting orgar	250

Testing_data

1	Body_ID	articleBody
2	0	A small meteorite crashed into a
3	4	Last week we hinted at what was to
4	5	(NEWSER) â€“ Wonder how long a
5	6	Posting photos of a gun-toting child
6	7	At least 25 suspected Boko Haram
7	8	There is so much fake stuff on the
8	9	(CNN) -- A meteorite crashed down
9	10	Move over, Netflix and Hulu.
10	11	Weâ€™ve all seen the traditional

Each Headline has an average of 15-25 words and article body has about 250 - 350 words.

Although our data is clean it is biased towards Unrelated stance. We can observe the distribution of classes in the following graph and table.



rows	unrelated	discuss	agree	disagree
49972	0.73131	0.17828	0.0736012	0.0168094

If we observe the table, we can see that unrelated class is dominating the data.

General Pre-processing: Pre-processing is essential to tune the data in a way suitable for our models. We started by converting all the text in our data to lowercase. Then we removed all the stopwords using nltk method, stopwords are not significant to find the relationship between headlines and bodies. After removing the stopwords from training and testing data we converted our headlines and bodies to form term frequency and inverse document frequency vectors(tf-idf), tf-idf is a popular method used to get numerical representation from text data.

Next, we transformed our tf-idf vectors so that they have maximum of 5000 features in headlines and bodies. Transform method selects the words which have high frequency. So, 5000 features from headlines and 5000 features from body tf-idf vectors. For some models we also used the Cosine similarity weights between the headlines and bodies as a feature. So, if we include Cosine similarity weights as a feature, we will have a total of 10001 features. This is general preprocessing procedure we followed for our models.

Doc2vec pre-processing: In this Doc2vec preprocessing case we did all the above preprocessing steps the only difference is here we converted our headlines and bodies data into vectors using Doc2vec method. Doc2vec is method which will extract the context of whole document rather than just the words. From this procedure we obtained 800 features.

Under sampling: To deal with the unbalanced data we decided to use the under-sampling technique. We wanted to give equal weights for all the classes. In our data the count of Agree class is 3678. We selected 3678 records randomly from all other 3 classes. Disagree class don't have 3678 records, so we had to repeat the records so that it will have same weight as the other classes. Using these records, we constructed tf-idf vectors for each class separately. For each class we have extracted 3000 features using transform method. Then we combined all the features (12000) from each class into a single matrix which we will pass into the model.

4 Model Component

Baseline Model: To evaluate the models we have written we need to establish a baseline first. To get the baseline we have used Random Forrest model with default parameters. It predicted all the records as the dominating class which is unrelated. We can observe the same from below

Confusion matrix obtained from testing predictions.

```
Confusion Matrix:
 [[    0     0     0  1903]
 [    0     0     0   697]
 [    0     0     0  4464]
 [    0     0     0 18349]]
              precision    recall  f1-score   support

          0       0.00      0.00      0.00      1903
          1       0.00      0.00      0.00       697
          2       0.00      0.00      0.00      4464
          3       0.72      1.00      0.84     18349

avg / total       0.52      0.72      0.61     25413
 Accuracy:  0.7220320308503522
```

Here 0 means Agree, 1 means Disagree, 2 means Discuss, and 3 means Unrelated. So, it predicted all the records as unrelated which gave an accuracy of 0.722. So as our data is unbalanced, we cannot evaluate the model based on the accuracy of the model. As we are dealing with multi class classification we decided to use Convolutional Neural Networks (CNN) and Multilayer perceptron (MLP) as our models.

CNN:

Convolution neural network is a class of neural networks which is mainly used for in analyzing visual imagery. It mainly works by learning the patterns in the images by forming multiple layered architecture. This technique was inspired from the biological processing of the humans in which the neurons in the resemble the organization of visual cortex.

CNNs in general needs less preprocessing when compared to other image classification methods. CNN in general consists of an input layer, output layer and multiple hidden layers. the hidden layers can be further divided into Convolution layers, pooling layers and fully connected layers.

How can we apply this to our NLP task at hand?

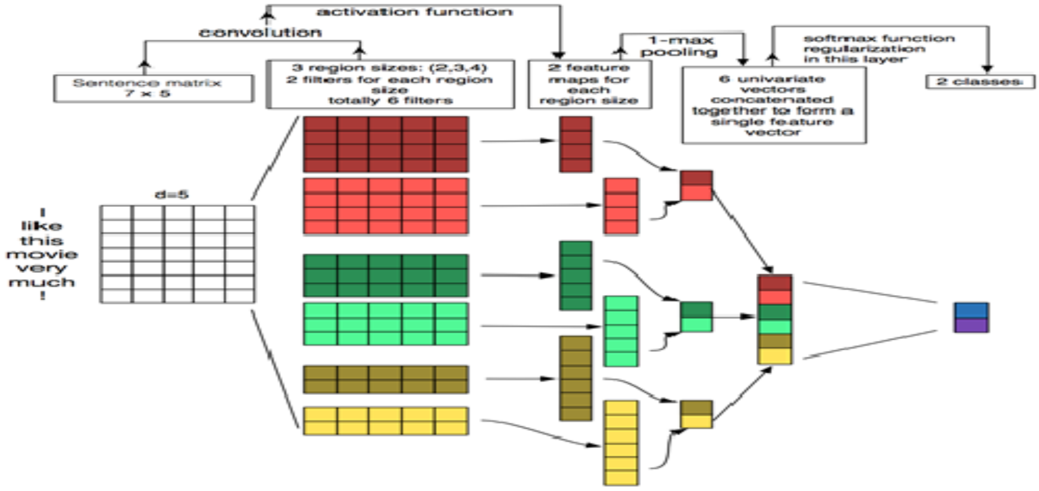
Instead of image pixels, the input to most NLP tasks are sentences or documents represented as a matrix. Each row of the matrix corresponds to one token, typically a word, but it could be a character. That is, each row is vector that represents a word. Typically, these vectors are word embeddings (low-dimensional representations) like word2vec or GloVe, but they could also be one-hot vectors that index the word into a vocabulary. For a 10-word sentence using a 100-dimensional embedding we would have a 10×100 matrix as our input. (Britz and Denny, 2015)

Parameters for our CNN:

In our CNN model we will be using two convolutional layers, for the first convolutional layer we have used 64 filters with each filter size of 4 we have followed up with relu activation combined with Max pool layer of size 2 after this for the second convolution layer we have used 32 filters of size 4 and followed up with relu activation combined with Max pool layer of size 2. High number of filters helps in increasing the learning rate. After second convolution layer we have used dense layer of size 512 with relu activation and for the final layer we have used a dense layer of size 4 with SoftMax activation. For the 2 convolution layers we have used dropout function to generalize the results.

MLP:

In this Multi-layer perceptron model, we will have 3 dense layers with a dropout function.



Imagesource:http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

5 Model results

CNN with tf-idf:

In this model we will give the pre-processed data obtained from tf-idf vectors as input to the CNN.

```
Confusion Matrix:
 [[    2     0     6  1895]
 [    1     0     3   693]
 [    3     0    13  4448]
 [   10     1    28 18310]]
              precision    recall  f1-score   support

          0       0.12      0.00      0.00      1903
          1       0.00      0.00      0.00       697
          2       0.26      0.00      0.01      4464
          3       0.72      1.00      0.84     18349

avg / total       0.58      0.72      0.61     25413
 Accuracy:  0.72108763231417
```

If we look at the confusion matrix from testing data, we can see that the model did not learn as well as we want. It did not classify the classes 0 -Agree, 1- Disagree, 2- Discuss very well. Its because of the unbalanced data. We have also included the cosine similarity weights along with he features obtained from tf-idf vectors in a model, the cosine weights did not help it couldn't predict the 0,1,2 classes very well.

CNN with Doc2vec:

In this model we have used the 800 features obtained from the Doc2vec preprocessing method.

```
Confusion Matrix:
 [[    1     0     6  1896]
 [    0     0     1   696]
 [    3     0     5  4456]
 [    5     1    17 18326]]
              precision    recall  f1-score   support

          0       0.11      0.00      0.00      1903
          1       0.00      0.00      0.00       697
          2       0.17      0.00      0.00      4464
          3       0.72      1.00      0.84     18349

avg / total       0.56      0.72      0.61     25413
 Accuracy:  0.7213630818872231
```

If we look at the confusion matrix, we can observe that the results are like the above method. This model also couldn't learn anything to predict the classes 0,1,2.

Multilayer perceptron with Under Sampling method:

In this model we have given features obtained from the under-sampling preprocessing method.

```
Confusion Matrix:
 [[ 633   18  268  984]
 [ 208    4  105  380]
 [1553   34  623 2254]
 [6304  135 2548 9362]]
              precision    recall  f1-score   support

          0       0.07      0.33      0.12      1903
          1       0.02      0.01      0.01       697
          2       0.18      0.14      0.16      4464
          3       0.72      0.51      0.60     18349

avg / total       0.56      0.42      0.47     25413
 Accuracy:  0.4179750521386692
```

Here the results are interesting if you look at the accuracy it is very low when compared to the baseline but as we said due to the unbalanced data it's not fair to look at the accuracy and evaluate the model. This model unlike other models we have showed before is able to predict the classes 0-Agrees, 1-Disagree, 2-Discuss classes very well. As the unrelated class is dominating the data the accuracy went down. These examples might help us answer why giving equal weights to all classes also gave us poor accuracy.

Sentence1: Mike is playing football with Ross.

Sentence2: Mike is not playing football with Ross.

Sentence3: Mike is not playing Tennis with Ross.

So, when we compare sentence 1 and sentence 2, we can clearly say that they both disagree with each other and sentence 3 is unrelated with sentence 1 and 2. But when we train our model as most of the words in sentence 1 and 2 are present in sentence 3 the model assumes that sentence 1 and 2 are also unrelated.

We also implemented CNN with same under sampling data and we achieved almost same results as MLP.

```
Confusion Matrix:
 [[ 509   30  439  925]
 [ 203    9  170  315]
 [1235   64 1053 2112]
 [4960  280 4206 8903]]
             precision    recall  f1-score   support

          0       0.07      0.27      0.12      1903
          1       0.02      0.01      0.02       697
          2       0.18      0.24      0.20      4464
          3       0.73      0.49      0.58     18349

avg / total       0.56      0.41      0.47     25413
 Accuracy:  0.4121512611655452
```

6 Conclusion

We have learned that unbalanced data effects the prediction performance of the models. If we look at the results of first two models, we can say that the models did not learn anything. But when we managed to give qual weights for all the classes the models ran using the Under-Sampling data gave better results. In the future we want implement Hierarchal prediction models by using 1 vs all classification. We also want to explore pre-processing techniques which might help us to identify more important features in the data.

References:

Uclmr. "Uclmr/Fakenewschallenge." GitHub, 18 May 2018, github.com/uclmr/fakenewschallenge.

Britz, Denny. "Understanding Convolutional Neural Networks for NLP." WildML, 10 Jan. 2016, www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/.

https://web.stanford.edu/class/cs224n/reports/2761239.pdf