Separate Factual & Non-Factual data in News articles

Dhruv Kaushik (MT18037), Subhani Shaik (MT18117), Palla Akhil Kumar (MT18130)

Computer Science Department

IIIT-Delhi, India

I. ABSTRACT

Newspaper articles are major sources of information about current affairs and events occurring throughout the world. News article usually contains a mixture of factual statements and non-factual statements. A fact can be considered as something that can be proven to be true. With respect to news articles, events which have actually happened and statements which can be proved to be true are considered as factual in nature, whereas non-factual data includes, opinions, beliefs, inferences, predictions, etc. The problem of separating factual data and non-factual data is considered as a classical NLP problem. We have developed a SVM classifier for classifying facts and non-facts in news article.

II. INTRODUCTION

A News article usually contains a mixture of factual data, like original news facts, and non-factual data, like writer's opinions, inferences, predictions, etc. We supposed that the factual information in a news article is the real unbiased information which a reader wants to retrieve from the news article. The goal of this project is to develop an Extractive Factual vs. Non-factual Information Retrieval system, which retrieves the original statements from a news article that are factual in nature. Various features that can be used for detecting factual content in news articles are studied in this project.

Factual information in news article is the real news which reader is usually interested in reading. Retrieving only the factual information from an article helps him to prevent reading irrelevant data present in the article. This retrieval can be useful in generating news headline or automatic summary of facts for a given news article.

With growth of internet, ample sources of news have come up over internet. Some are commercial news channels, public organizations, or even individual based sources. Different sources produce different quality of contents. Determining the amount of factual information in a news article can help in checking its content quality. This is also helpful in News Articles information retrieval systems, to rank higher those news articles having more factual content than remaining for the same topic.

This can be used for designing systems that perform automatic credibility check of news article by retrieving factual claims made by it and processing them using multiple sources.

III. LITERATURE SURVEY

Hong Yu, Vasileios [1] created document level classifier to label entire document as factual or non-factual in first part of their research, using dataset of 173,252 Wall Street Journal. They used token-based features like unigrams, bigrams and trigrams along with syntactic features like Part Of Speech (POS) in each sentence. A. Stepinski and V. Mittal [2] also developed classifier to classify complete news article as fact or opinion. It categorizes each test sentence as factual or opinion using Passive Aggressive Algorithm trained on token based features (n-grams) features. Based on the scores of all sentences in an article, it categorizes the article. Document-level classifier can categorize document as a whole as subjective text (non-factual) or objective texts (factual). But in reality, news article usually consists of a mixture of them. Wiebe et al. [3] observed that around 44% of sentences in a news collection were found to be subjective. Ishan Sahu [4] designed sentence level classifier for classifying text sentence in news article as factual and non-factual. N-grams, part of speech tags and entity types were used as features for the classification of sentence.

IV. DATASET

We dataset we used is MPQA (Opinion) Corpus [5] version 1.2. The MPQA Opinion Corpus contains 535 news articles which are manually annotated for phrases containing opinions and other subjective states like inferences from facts, sentiments, beliefs, emotions, predictions, etc.). We parsed the annotations, marked in MPRA format, to retrieve information about subjective nature of phrases in sentences, so as to label sentences as factual or non-factual.

V. FEATURE EXTRACTION

Every statement in corpus is taken as a data point. The following features are extracted for each data point to form its feature vector:

1. SVM Feature Selection

A. N-grams extraction

We generated Unigrams, Bigrams and Trigrams sets each for Factual and Non-Factual training data individually. We computed these six features using them :

a. **average_Factual_unigram** score: no. of unigrams present in the sentence as well as in factual unigram set divided by no. of unigrams in sentence.

b. **average_Factual_bigram** score: no. of bigrams present in the sentence as well as in factual bigram set divided by no. of bigrams in sentence.

c. **average_Factual_trigram** score: no. of trigrams present in the sentence as well as in factual trigram set divided by no. of trigrams in sentence.

d. **average_NonFactual_unigram** score: no. of unigrams present in the sentence as well as in nonfactual unigram set divided by no. of unigrams in sentence.

e. **average_NonFactual_bigram** score: no. of bigrams present in the sentence as well as in nonfactual bigram set divided by no. of bigrams in sentence.

f. **average_NonFactual_trigram** score: no. of trigrams present in the sentence as well as in nonfactual trigram set divided by no. of trigrams in sentence.

B. Part-of-speech (POS) tags extraction

We generated the POS tags for whole corpora by considering it as single string. Later we found list of all distinct POS tags among them and used the distinct POS tags as feature dimensions for our vector. For feature extraction on this feature, we generated POS tags for each sentence and filled the dimension values for respective POS tags in the sentence feature vector. We used Spacy library for computing both POS tags and Named Entities (which is used in Entity type extraction) in a given statement.

C. Entity types extraction

Entities in a sentence can be very helpful in predicting class of the text. In factual sentences the entities like money, number, time, date etc. types will be more, which can help us in differentiating the factual and non-factual sentences. We have extracted all the unique entities present in the training data and each of this is considered as a dimension. For each sentence, while extracting this features, we find all entities in it and for each entity occurring in the sentence, we add the count of occurrence of the entity in its corresponding dimension in feature vector of the sentence.

D. Subjective Patterns(TPattern) Extraction

The non-factual part of the sentence can be characterized by simple POS patterns denoting some kind of opinion. Patterns containing adjective or adverb with contextual words are used for detecting semantic orientation. We will use 5 cases and each will be a dimension and count is added in that corresponding dimension.

	First Word	Second Word	Third Word (Not Extracted)
1	JJ	NN or NNS	anything
2	RB, RBR, or RBS	JJ	not NN nor NNS
3	JJ	JJ	not NN nor NNS
4	NN or NNS	JJ	not NN nor NNS
5	RB, RBR, or RBS	VB, VBD, VBN, or VBG	anything

E. Sentiment Extraction

To decide the sentence or phrase for fact or non-fact we also computed sentiment score of the sentence using Stanford coreNLP libraray. It gives us scores from 0 to 4 (0 for ‘very negative’, 1 for ‘negative’, 2 for ‘neutral’, 3 for ‘positive’, 4 for ‘very-positive’). This score also adds one dimension in our feature vector.

F. POS Pattern Extraction

We generated separate sets of Trigrams and Four-grams of POS Tags, each for Factual and Non-Factual training data, using which these 4 features were computed :

a. **avg_factual_pos_pattern_3grams** score: no. of POS trigrams present in the sentence as well as in

factual POS trigrams set divided by total number of trigrams in sentence POS tag.

b. **avg_factual_pos_pattern_4grams** score: no. of POS fourgrams present in the sentence as well as in factual POS fourgrams set divided by total number of trigrams in sentence POS tag.

c.**avg_nonfactual_pos_pattern_3grams** score: no. of POS trigrams present in the sentence as well as in non-factual POS trigrams set divided by total number of trigrams in sentence POS tag.

d. **avg_nonfactual_pos_pattern_4grams** score: no. of POS fourgrams present in the sentence as well as in non-factual POS fourgrams set divided by total number of trigrams in sentence POS tag.

2. Features for LSTM and GRU:

The feature vector representing a statement (data point) used for LSTM is a list of word embeddings for all tokens in the statement. The word embedding for a statement is obtained by using Word2vec module of Gensim library, trained over Wikipedia articles corpus.

Word embedding for each token is of fixed length 300. But length of the sentences are different, varying from 1 to 140. So, in order to bring the feature vector of each sentence to same dimension, dummy word embedding, containing all zeroes, are padded to bring feature vector size to 140 (feature vector shape 140x300).

VI. TECHNIQUES

A. Support Vector Machine (SVM)

We used Linear Support Vector Machine (SVM) model as our classifier. The hyper parameters for this model are tuned using GridSearch with cross-validation of 5 folds.

B. Long-Short Term Memory (LSTM)

It is Recurrent Neural Network(RNN) along with some memory to hold the contextual information in sentence.

We used several variants of LSTM like 2 layers, 3 layers, with dropout rate, without dropout, cutting down the maximum length of sentence (ignoring some unnecessary sentences) and taking average of the word embeddings.

We trained on these architectures by setting parameters learning rate of 0.0001, epochs=10, Adam optimizer technique and batch size of 16, dropout rate=0.2.

C. Gated Recurrent Unit (GRU)

It is Recurrent Neural Network(RNN) along with some memory to hold the contextual information in sentence. It has less parameters compared to LSTM and is very light. We used GRU with 3 layers along with dropout rate.

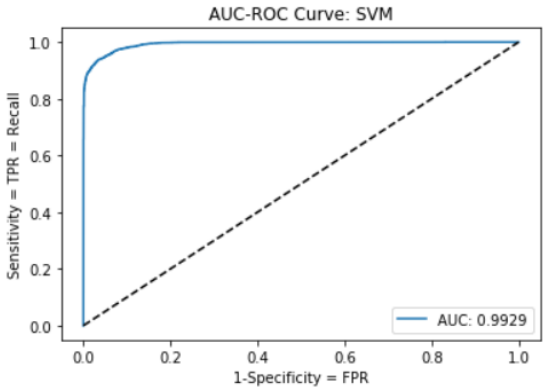
We trained on these architectures by setting parameters learning rate of 0.0001, epochs=30, Adam optimizer technique and batch size of 16, dropout rate=0.2.

VII. Results and Analysis

A. SVM

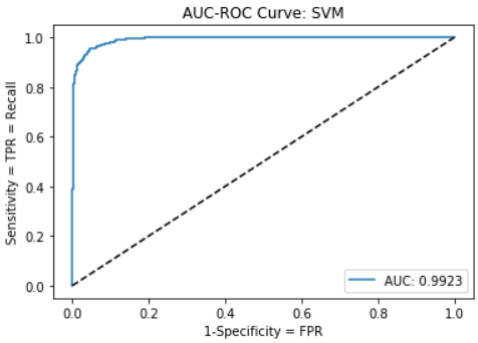
Train Accuracy and ROC:

Evaluation Metric	Score
Accuracy	0.9525
Recall	0.9394
Precision	0.9546
F1 score	0.947



Test Accuracy and ROC:

Evaluation Metric	Score
Accuracy	0.951
Recall	0.9438
Precision	0.9448
F1 score	0.9443



We got a good precision and recall of 94.47% and

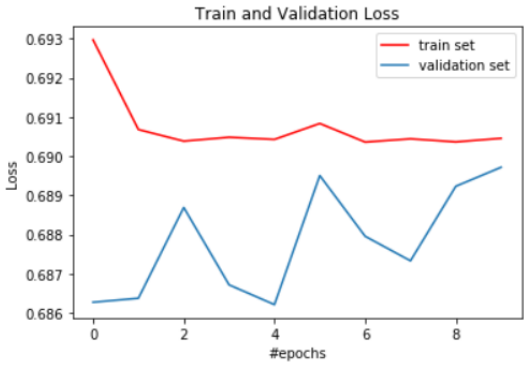
94.3% respectively on hold out set. Our training precision and recall is 95.4%, 93.9% respectively which indicated that our training data is representative of the data and model is not over fitted.

B. LSTM and GRU

Model	Train Accuracy %	Validation Accuracy %
LSTM-2 layers with dropout	54.89	55.96
LSTM-2 layers without dropout	56.7	57.22
LSTM-3 layers with dropout	54.75	55.96
LSTM-3 layers without dropout	54.92	55.31
LSTM-3 layers with sentence max length	55.44	53.78
LSTM-3 layers without dropout and average features	54.90	55.95
GRU-3 layers	55.65	59.87

LSTM:

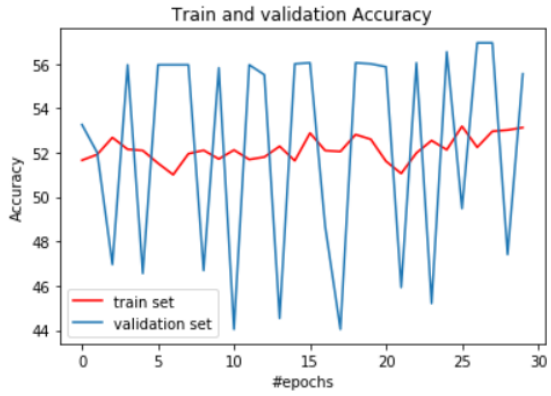
Train and Validation set Loss:



GRU:

Train and Validation set Loss:

Train and Validation set Accuracy:



In LSTM by increasing epochs it will get better results but after some epoch the values remains

constant. We tried averaging the word embedding and it is not performing good than general architectures.

In GRU, we used word embedding as input and it is working well than LSTM because it have less number of parameters and loosely constrained.

VIII. CONCLUSION

The Linear SVM classifier model developed in this project performs this task efficiently by classifying all the factual and the non-factual statements in a given news article with good accuracy. It works more accurately over short or medium-sized statements than long statements, as long statement usually contains both subjective as well as objective type of phrases. This increases the probability of misclassification of long statements. Reducing the size of the unit of classification from whole statement to ngrams phrase may help in overcoming this limitation.

LSTM and GRU don't performed well. This could be because these technique will perform based on network weights and don't considers the some important features which we done in classical extractive techniques. These techniques are mainly depends on dataset like size of the dataset, representation of dataset etc. We used very less dataset, which is not enough to learn from that dataset. This also depends on parameters like learning rate, number of epochs etc.

The application of this Information retrieval system can be extended to retrieval of factual information from Research paper, journal article or online web article.

IX. REFERENCES

1. Hong Yu, Vasileios Hatzivassiloglou (2003). Towards Answering Opinion Questions: Separating Facts from Opinions and Identifying the Polarity of Opinion Sentences. EMNLP '03 Proceedings of the 2003 conference on Empirical methods in natural language processing. Pages 129-136

2. A. Stepinski and V. Mittal. A fact/opinion classier for news articles. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07, pages

807{808, New York, NY, USA, 2007. ACM.

3. Wiebe, T. Wilson, and M. Bell. 2001. Identifying Collocations for Recognizing Opinions. In Proceedings of the ACL-01 Workshop on Collocation: Computational Extraction, Analysis, and Exploitation.

4. Ishan Sahu, Debapriyo Majumdar (2017). Detecting Factual and Non-Factual Content in News Articles. Proceeding in CODS '17 Proceedings of the Fourth ACM IKDD Conferences on Data Sciences. Article No.17

5.http://mpqa.cs.pitt.ed/

6. P. D. Turney. Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02, pages 417{424, Stroudsburg, PA, USA, 2002.