Towards Automated Fact Checking

A thesis submitted for
B. Tech. Project
in
Computer Science and Engineering

by

Prabhat Agarwal (13CS10060)
Priyank Palod (13CS30046)

advised by

Dr. Pawan Goyal
Dr. Saurabh Bagchi

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

April 2017

Certificate

This is to certify that the thesis titled **Towards Automated Fact Checking** submitted by **Prabhat Agarwal (13CS10060)** to the Department of Computer Science and Engineering is a bona fide record of work carried out by him under my supervision and guidance.

Dr. Pawan Goyal
Assistant Professor
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
April 2017

Declaration

I, **Prabhat Agarwal** hereby certify that this thesis titled **Towards Automated Fact Checking**

- Is an original work and has been done by me under the guidance of my supervisor;
- The work has not been submitted to any other Institute for any degree or diploma;
- While writing the thesis I have conformed to norms and guidelines given in the Ethical Code of Conduct of the Institute;
- Whenever I have used materials (data, model, figures and text) from other sources, I have given due credit to them by citing them in the text of the report, giving their details in the references, and following fair use doctrine policies of copy righted materials if any used in this thesis.

Prabhat Agarwal

2nd April 2017

Abstract

With the rise of technology, social media and new forms of journalism, it is now easier than ever to disseminate falsehoods and half-truths, faster than the rate at which human fact-checkers can expose them. Automation may hold the key to far more effective and efficient fact-checking. A fully automated fact checker would need advanced artificial intelligence and the problem of building the holy grail of computational fact checking seems intractable at this point of time. But in pursuing this ambitious goal, we can help to improve fact-checking and the political discourse by taking baby steps in this direction. We have tried to solve two such steps namely extraction of check-worthy claims and stance detection of a claim with other news articles/facts.

In our quest to automate the process of extraction of claims to fact check, we have used the presidential debates of U.S. Election 2016. We have achieved an F1-score of 0.224 which is comparable to the performance of current state-of-the-art "ClaimBuster" while generating better ranking for check-worthy claims (NDCG@100 of 0.224 as compared to NDCG@100 of 0.184 for ClaimBuster).

The stance detection task utilizes the dataset which is a part of Fake News Challenge 2016 (FNC1). The challenge has two parts - Task A, which is the classification of headline and article pair as related/unrelated and Task B, classifying every related pair of headline and article by their stance into "agree", "disagree" or "discuss" class. Our system performs with an accuracy of 96.35% on task A (as opposed to 95.61% using the official baseline approach) and with an accuracy of 74.52% on task B as opposed to 67.68% using the official baseline approach).

Contents

Chapter 1

Introduction

Fact checking is the act of checking factual assertions in a non-fictional text in order to determine the veracity and correctness of the factual statements in the text. In the digital era, fact checking has been extended to include public verbal statements, and interviews made by public figures such as politicians, pundits, etc. It is commonly performed by journalists employed by news organizations in the process of news article creation. More recently, institutes and websites dedicated to this cause have emerged such as factcheck.org and PolitiFact, etc. Some examples of fact-checked statements, together with the verdicts offered by the journalists are shown below.

Example 1: *Clinton's Equal Pay Claim* (*politifact.org*[1]): Clinton said Donald Trump "*doesn't believe in equal pay.*"

Verdict: Trump has said pay should be based on performance, not gender – so he does appear to favor uniform payment if performance is alike. Clinton's statement is partially accurate but leaves out important details or takes things out of context. Hence PolitiFact rated it as *Half True*.

Example 2: *Trump's Claim on immigrants and crime* (*politifact.org*[2]):

[1]http://www.politifact.com/truth-o-meter/statements/2016/nov/02/hillary-clinton/hillary-clinton-says-donald-trump-doesnt-believe-e/

[2]http://www.politifact.com/truth-o-meter/statements/2016/nov/03/donald-

Donald Trump said "*Thousands of Americans have been killed by illegal immigrants.*"

Verdict: Trump puts no timeframe on his comment, leaving his audience to fill in the blanks. In reality, there is no solid data for homicides committed by people living illegally. His implicit suggestion is that people should fear illegal immigrants more than citizens, but we don't see evidence for that. Research shows immigrants are less likely to engage in criminal behavior than the native-born population. However, like the legal U.S. population, some of the country's more than 11 million undocumented immigrants have committed murders. Trump's statement was open-ended enough that Politifact rated it *Half True.*

The growing movement of political fact-checking plays an important role in increasing democratic accountability and improving political discourse [16, 29]. A recent study [43] showed that:"By and large, citizens heed factual information, even when such information challenges their partisan and ideological commitments". Politicians and media figures make claims about "facts all the time, but the new army of fact-checkers can often expose claims that are false, exaggerated or half-truths. The number of active fact-checking websites has grown from 64 a year ago to 96 in 2016, according to the Duke Reporters' Lab (fig.1.1). [3]

With the movement towards accountability and transparency, the amount of data available in the public domain is ever increasing. Such data open up endless possibilities for empowering journalism's watchdog function - to hold governments, corporations, and powerful individuals accountable to society. Though there has been an increase in the cadre of investigative journalists (fig 1.1), we are facing a widening divide between the amount of data available

trump/trump-leaves-out-context-claim-about-immigrants-an/

[3]http://reporterslab.org/global-fact-checking-up-50-percent/

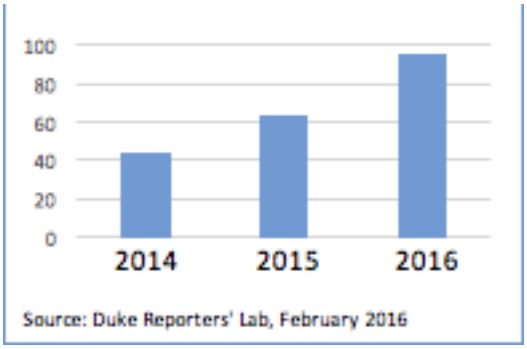


Figure 1.1: Active fact-checking sites around the world

and its efficient usage in investigative journalism.

1.1 Motivation for Automation

Fact checking is a very difficult and time-consuming task for journalists. Traditional fact-checking experts cannot keep up with the enormous volume of information that is now generated online. In fact, just finding the claims to check is a big task. Journalists have to spend hours going through the transcripts of speeches, debates, interviews to identify the claims they need to research. This creates a significant gap between the moment a politician makes a statement and when the fact-check is ultimately published. A complicated fact-check may require two or more days, while on the other hand, a meme typically flows from news media to blogs in just 2.5 hours [23]. The voters usually do not get the information when they really need it. This is one of the several factors that emboldens politicians to keep repeating claims even when they are false.

Computation may hold the key to far more effective and efficient fact-checking, as Cohen et al. [8, 9] have pointed out. Our eternal quest is a completely automatic fact-checking platform that can detect a claim as it appears in real time, and instantly provide the public with a rating about its

accuracy. It makes its calls by consulting databases of already checked claims, and by analyzing relevant data from reputable sources. In this project, we explore the technical challenges we will face in automating fact-checking and potential solutions for them.

Fact checking organizations may not arrive at a consensus regarding accuracy. Research support the notion that more than one such fact checking source needs be consulted, to arrive at a consensus of opinion on statements being checked [10]. Computational fact checking may significantly enhance our ability to evaluate the veracity of dubious information. It will also eliminate the problems related to human nature like bias.

This would lead to a world with true democracy and freedom. People would know their politicians better and hence will elect better leaders.

1.2 Problem Definition

The task of fact checking can be defined as the assignment of truth value to a claim made in a particular context. It might naturally seem that this is a binary classification task. However, it is often the case that statements are not completely true or completely false. For instance, consider the statement in example 2. This claim has been rated as half true due to various reasons as mentioned in the verdict. Therefore it is better to consider fact-checking as an ordinal classification task [15], thus allowing systems to capture the nuances of the task.

The ultimate goal of the problem is to build a fully automated fact checker that can perform just as well as a team of journalists, if not better. This fully automated system, sometimes referred to as "The Holy Grail" in research literature [19], will bear the following characteristics:

Fully Automated: It checks facts without human intervention. It takes

as input the video/audio signals and texts of a political discourse and returns factual claims and a truth rating for each claim.

Instant: It immediately reaches conclusions and returns results after claims are made, without noticeable delays.

Accurate: It is equally or more accurate than any human fact-checker.

Accountable: It self-documents its data sources and analysis, and makes the process of each fact-check transparent. This process can then be independently verified, critiqued, improved, and even extended to other situations.

A fully automated fact-checker calls for fundamental breakthroughs in multiple fronts and, eventually, it represents a form of Artificial Intelligence (AI). Such a system mandates many complex steps - extracting natural language sentences from textual/audio sources; separating factual claims from opinions, beliefs, hyperboles, questions, and so on; detecting topics of factual claims and discerning which are the check-worthy claims; assessing the veracity of such claims, which itself requires collecting information and data, analyzing claims, matching claims with evidence, and presenting conclusions and explanations. Each step is full of challenges. The quest for "The Holy Grail" will constantly drive us to improve this important journalistic activity.

1.3 Challenges

There are several challenges in automated fact-checking. One of the major challenges is that fact checking requires world knowledge and spatial and temporal context of a fact candidate. The world knowledge needs to be updated in real time. Also, the sheer vast amount of knowledge and context required makes general fact checkers a thing of remote future. Therefore, more work is concentrated on building fact checkers that work in particular domain, for example in the domain of American politics. But it is still very difficult to

find the correct sources from where we can get the information relevant to a statement.

Fact checking often requires calculations and analysis of the data. Hence, it is not just an information retrieval task, it requires a lot of artificial intelligence, making it extremely challenging. While some calculations can be as simple as computing percentages from data, others might be a lot more complicated.

Another significant challenge comes from the inherent ambiguity of natural languages. There can be multiple interpretations of the same statement leading to different truth values. In fact, in many cases, there is a deliberate deception. Politicians often try to hide the facts by omitting the context. Also, sometimes even factually correct statements are used to convey incorrect implicit messages. For instance, in example 2, Trump says "Thousands of Americans have killed by immigrants" which is factually correct. But the underlying implicit message that he wants to convey is that illegal immigrants are dangerous. While it is true that thousands of Americans have been killed by immigrants, a notable point is that the same crime rate exists in the legal US population too, if not more. Hence, the message being sent is wrong in spite of the correctness of claim. This suggests that fact-checking should also include generating counter-arguments for a claim as if it is debating on the claim. This requires a system too intelligent to be possible in near future.

1.4 Broad Approach

Once we have some text data to fact-check, the approach can be broadly divided into the following steps (fig.1.2):

- **Claim Extraction**: Extracting the checkable and check-worthy facts from the text.

- **Fact builder**: Understanding the semantics of the fact statements and building their structural representations. It involves resolving references, removing intentional/unintentional vagueness of the facts.

- **Source Identification**: Identification of reliable sources of information needed for the fact-check using the contextual and semantic information of the fact. These sources can be some news websites, Wikipedia, data available on government websites or personal web pages and blogs. It also involves corroborating data from multiple sources with regards to its quality and completeness. It in turn requires determining the stance of the sources with respect to the fact candidate.

- **Information Analysis**: Extraction of relevant information from the sources identified above and analyzing it.

- **Ordinal Classifier**: Reaching a verdict from the retrieved data and also generating suitable arguments with references to support the verdict.

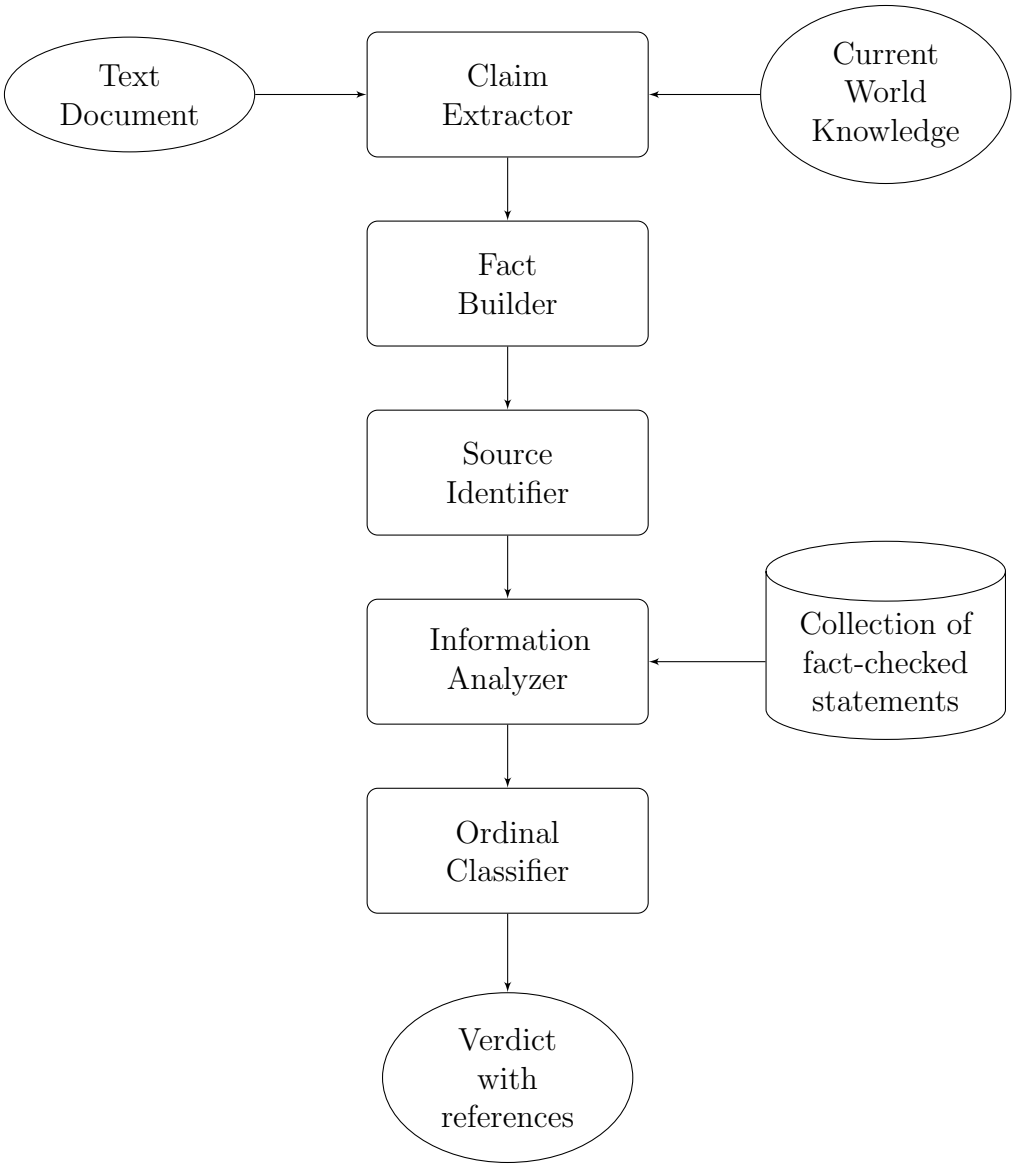


Figure 1.2: Modules of automated fact checking system

Chapter 2

Related Works

The field of automated fact-checking is relatively new. With the rise of fake news in mainstream media, there has been a rise in interest in the research community to build an automated fact checking system. Automating the process of fact checking has recently been discussed in the context of computational journalism [8, 14]. Vlachos and Riedel [39] introduced the task of fact checking and presented a dataset of factual claims collected from PolitiFact.com and Channel4.com. They also discussed some baseline approaches for the task and the challenges that need to be addressed.

Wu et al. [44] provided a modeling framework for fact-checking. They modeled a claim as a parameterized query over data, whose result would vary as its parameter setting is changed. They also showed how different fact-checking tasks, such as finding counterexamples and reverse-engineering vague claims, can be modeled using the framework. Their work provides a starting step for the Information Analyzer of the automated fact-checking system described above.

Nakashole and Mitchell [28] provide a language based approach for assessing the truth of doubtful or debated statements in Web contents using a combination of objectivity and co-mention influence. Ciampaglia et. al. [7]

describes a method for accessing the truth of simple statements of fact using a large-scale knowledge graph obtained from Wikipedia.

To the best of our knowledge, no prior study has focused on a full end-to-end automated system for fact checking. A relevant work focusing on claim extraction is by Hassan et. al. [18, 19]. They present the system ClaimBuster which ranks sentences based on their check-worthiness (a factual claim whose truthfulness is important to the public). It forms the first module in our proposed system. The system has many shortcomings and hence we explore the task further to overcome the shortcomings.

Another important building block in the third module of our proposed system is detecting stance between two text pairs to better understand the information available from different sources on the web. Over the last decade, there has been active research in modeling stance. Many people have worked in detecting stance in congressional debates [38] or debates in online forums [36, 27, 11, 40, 17, 37]. The SemEval 2016 Task 6 [26] focused on detecting stance in tweets, relative to a given known/unknown topic. However most of these works focus on target-specific stance prediction and hence are different from detecting stance between text pairs.

The task of stance detection between text pairs shares a lot in common with various natural language processing tasks, such as natural language inference [5, 41] and machine comprehension [20, 6], but is different in its own aspects due to domain and objective.

A relevant work on stance detection between text pairs, particularly news headline pairs was by Ferriera et. al. [12]. They used hand-crafted features of text pairs like bag of words and n-grams matching for stance classification on the Emergent dataset [35]. We further explore this task on the fake-news challenge dataset which is also derived from Emergent dataset [35, 12].

Chapter 3

Extracting Check-worthy Claims

We have worked on the first stage of the pipeline (fig. 1.2), that is, extracting the checkable and check-worthy claims from the text. Our system classifies sentences as check-worthy/not-check-worthy and produces a ranking of the sentences on the basis of its check-worthiness. The ranking scores help prioritize their efforts in assessing the veracity of claims. It also helps in producing a fact check of a text in order of significance so that a reader can see fact-check of more interesting claims first.

It is natural that check-worthiness is heavily dependent on the context of the text.

We describe our system in the nest sections. Section 3.1 describes the dataset and how it was constructed. Section 3.2 describes the features used in our classifier and section 3.3 studies the importance of the features. Section 3.4, 3.5 and 3.6 presents the evaluation of the system and comparison with ClaimBuster [19][1].

[1]http://idir-server2.uta.edu/claimbuster

3.1 Dataset Construction

To train the classifier for classifying sentences as check-worthy or not we required a labeled dataset of sentences. Creating a labeled dataset by human annotation has two shortcomings: firstly it requires a lot of human labor and time, and secondly it induces human bias in what is considered check-worthy. Hence we sought a different approach to creating the dataset.

We constructed a dataset of sentences spoken by presidential candidates in all presidential debates in the primary elections 2016 held between August 2015 and April 2016. The debate transcripts were crawled from CNN's website[2] using scrapy[3]. We extracted 15235 sentences from the 16 debates using Stanford's CoreNLP tool [25] after filtering out sentences spoken by speakers other than the candidates.

We used the facts checked by different fact-checking organizations such as Politifact[4], factcheck.org, New York Times[5], Washington Post[6], NBC news[7], etc. For each debate in our dataset, we extract the facts from the debate checked by these organizations and label their union as check-worthy. We hypothesize that taking the union will reduce the bias that the organizations may be having [10]. It gives us less manually intensive and less bias way to generate the dataset.

The fact checked by any of the organizations is first manually located in the debate and then the sentence corresponding to the fact is labeled as check-worthy. If the fact checked spans multiple sentences we label all those sentences as check-worthy. It is a manually intensive task yet less intensive than labeling all 15235 sentences. In this way out of 15235 sentences, 608 sentences were

[2]http://transcripts.cnn.com/TRANSCRIPTS/se.html
[3]https://scrapy.org/
[4]http://www.politifact.com/
[5]http://www.nytimes.com/
[6]https://www.washingtonpost.com/
[7]http://www.nbcnews.com/

labeled as check-worthy, and consider the rest 14627 sentences as not check-worthy.

3.1.1 Errors in the Dataset

Since the sentences in the not-check-worthy class were not manually annotated, there is some error in the dataset. Due estimate the error and find possible sources of error, we took a random sample of 50 sentences in the not-check-worthy class and manually annotated them. Out of the 50, 5 was labeled as check-worthy in the process. Hence the dataset has an estimated error of 10% in the labels of the not-check-worthy class. Further investigation in the possible sources of error revealed the following:

- The facts that have already been checked previously is mostly not checked again by any fact-checking organization and hence are labeled as not-check-worthy though they are very check-worthy. Out of the 5 mislabeled sentences we took, 2 were already checked by some fact-checking organization.
- Often, facts are checked either because they are very hard to check or because of time and resource constraints.

3.2 Features

We considered different combination of the following features for our classification:

- **POS tags (POS)**: Value of each tag is the number of times the POS-tag appears in the sentence. There is a total of 45 such POS-tags.

- **Frequent Bigrams (B)**: We extracted all the bigrams which occur more than 5 times in the sentences labeled as check-worthy. We removed the bigrams consisting of only stop words and finally had a total of 60 bigrams extracted. We use the presence of each bigram as a feature - the value is 1/0 if the bigram is present/absent in the sentence.

- **Subjectivity lexicon (S)**: The count of strongly and weakly subjective words in the statement is taken as features. The subjectivity of words is taken from a lexicon due to Wiebe and Mihalcea, 2006 [42]. [2 features]

- **Topic Distribution (T)**: We trained LDA [3, 33] on all previous debates[8] from the year 1960 and from the years 1976 to 2012, for 20 topics. For each sentence, we use the trained model to get the distribution of topics in it, and these score as features. We also make all scores less than 0.0001 to 0.

- **Dependencies (D)**: Some analysis of the structure of the data revealed that certain dependencies like compound, case, amod, det etc. appear more frequently in the checked claims than in the unchecked ones. We took 6 of these dependencies and used their presence as a feature.

- **Word Embeddings (E)**: We used the mean of the embeddings of the non-function words in the sentence to get the vector for the sentence. We used pre-trained vectors trained on part of Google News dataset (about 100 billion words). The model contains 300-dimensional vectors for 3 million words and phrases.[9]

- **Length (L)**: We used the number of tokens in the sentence as a feature. Shorter sentences generally are supporting utterances in speech

[8] http://www.presidency.ucsb.edu/debates.php
[9] https://code.google.com/archive/p/word2vec/

and seldom has significance on its own. Hence shorter sentences tend to be non-check-worthy (fig. 3.1).

- **Sentiment (Sm)**: We used NLTK's VADER [21] sentiment system to obtain the polarity of each sentence. Hence we get 4 features corresponding to the each of the polarity: positive, negative, neutral and compound. The polarity scores are a good indicator of the subjectivity of a sentence.

- **Entity Type (ET)**: We use Stanford's named entity recognizer [13] to obtain the named entities of each class in the sentence and use their count as a feature. Presence of named entity of different classes are indicative of important point being discussed. For example, a sentence containing entity of type number or person is more likely to be check-worthy. [12 features]

- **Homogeneity of the sentence (HOM)**: A sentence co-occurring with similar sentences is indicative of a coherent discourse which is emphasized by the speaker. Hence to take such global relations into account, we calculate how similar a sentence is with its neighbors. We calculate the tf-idf vector for each sentence and take the maximum similarity of the sentence within its block (A block is a group of 4-5 sentences spoken together by the same speaker as retrieved from the debate transcript). [1 feature]

- **LIWC Category (LIWC)**: We consider the LIWC [30, 31] categories: feeling, perception, certainty, and time, and count the number of words of each category in the sentence and use that as a feature. [4 features]

- **Verb Category (VC)**: We use five categories for the verbs of the sentence: reporting (e.g. say, told), perception (e.g. feel), belief (e.g. think), knowledge (e.g. admit), and doubt (e.g. doubt, hope). Factual sentences

Figure 3.1: Probability distribution of sentence length in check-worthy and not check-worthy sentences.

tend to have reporting or knowledge verbs whereas presence of doubt, perception verbs indicate opinion. Hence we use the number of verbs in each category as a feature. [5 features]

We now present a brief discussion of the relevance of some of the features for this task:

3.2.1 POS tags

POS tags contain some important characteristics which are useful to determine the check-worthiness of a sentence. Some of these are:

- Presence of cardinal tag: Presence of a cardinal is an important feature for check-worthiness of claims. The statements involving numbers are usually some concrete facts giving some important statistical information. This makes a statement with numbers being more probable to be check-worthy.

- The tense: The POS tags give us the information of the tense of the statements. Tense is important as check-worthy statements are usually in the past or present tense. The use of future tense denotes that the sentence is more of a promise or a prediction, and these type of sentences are usually more difficult to check.

- Other POS tags may also be very important. For example, the pos tag for the $ sign seems to be important as its presence denotes that the statement is a monetary claim and hence may be important to fact-check.

3.2.2 Bigrams and Word embeddings

Bigrams and embeddings contain the contextual information needed to identify the check-worthy claims. For example:

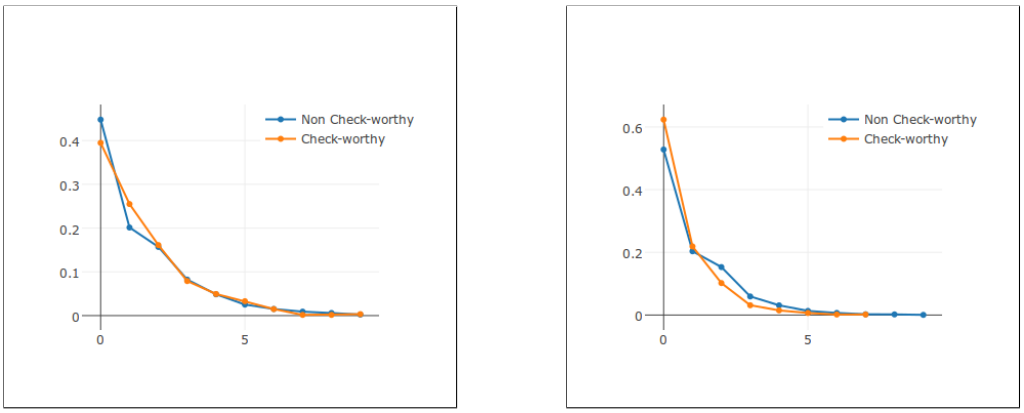
- Some bigrams like ‘wall street’, ‘this country’, ‘president Obama’, ‘to pay’, ‘the last’, ‘the highest’, ‘the fact’ etc. come a lot more often in check-worthy sentences than in non check-worthy ones.

- Similarly, embeddings are important as they represent the sentence itself

in the semantic space. They capture important semantic information and hence can capture the important topics.

3.2.3 Count of subjective words

The Subjectivity of the words in a sentence plays an important role in the classification. Presence of subjective words like 'admire', 'absurd', 'amusing', 'sinful', 'sincerely' etc. indicates that the claim might be less likely to be check-worthy. Hence, we took the count of subjective words in the sentence as a feature. Figure 3.2 shows the probability distributions of the count of weakly and strongly subjective words in the two classes.



(a) Probability distribution of the count of weakly subjective words.

(b) Probability distribution of the count of strongly subjective words.

Figure 3.2: Importance of Subjective Words for the task.

3.2.4 Topic Distribution

Claims in certain topics, like monetary policies, education and health care are more probable to be check-worthy. While claims pertaining to topics like the family and personal life of the individual might not be so check-worthy. To capture this information, we trained a Gibbs LDA on debates from 1960 to

2012 to get the top 20 topics in the American political scenario. Using debates as old as 1960s help in getting evergreen topics, not just the temporary ones.

3.2.5 Dependencies

On some experimentation and analysis on the training data, we found out that certain dependencies occur more frequently in checked statements then in the unchecked ones. We took 6 of these - namely "compound", "case", "nummod", "amod", "det", "punct" as our features.

3.3 Feature Importance

We trained an extreme random forest classifier for which we used GINI index to measure the importance of features in constructing each decision tree. The overall importance of a feature is its average importance over all the trees. Figure 3.3 shows the importance of features. The most important feature is pos_CD. This is not surprising as the statements with cardinals are usually more factual and check-worthy. This is also the reason for the high importance of nummod dependencies. pos_VBD is also important as it tells that the statement is in past tense.

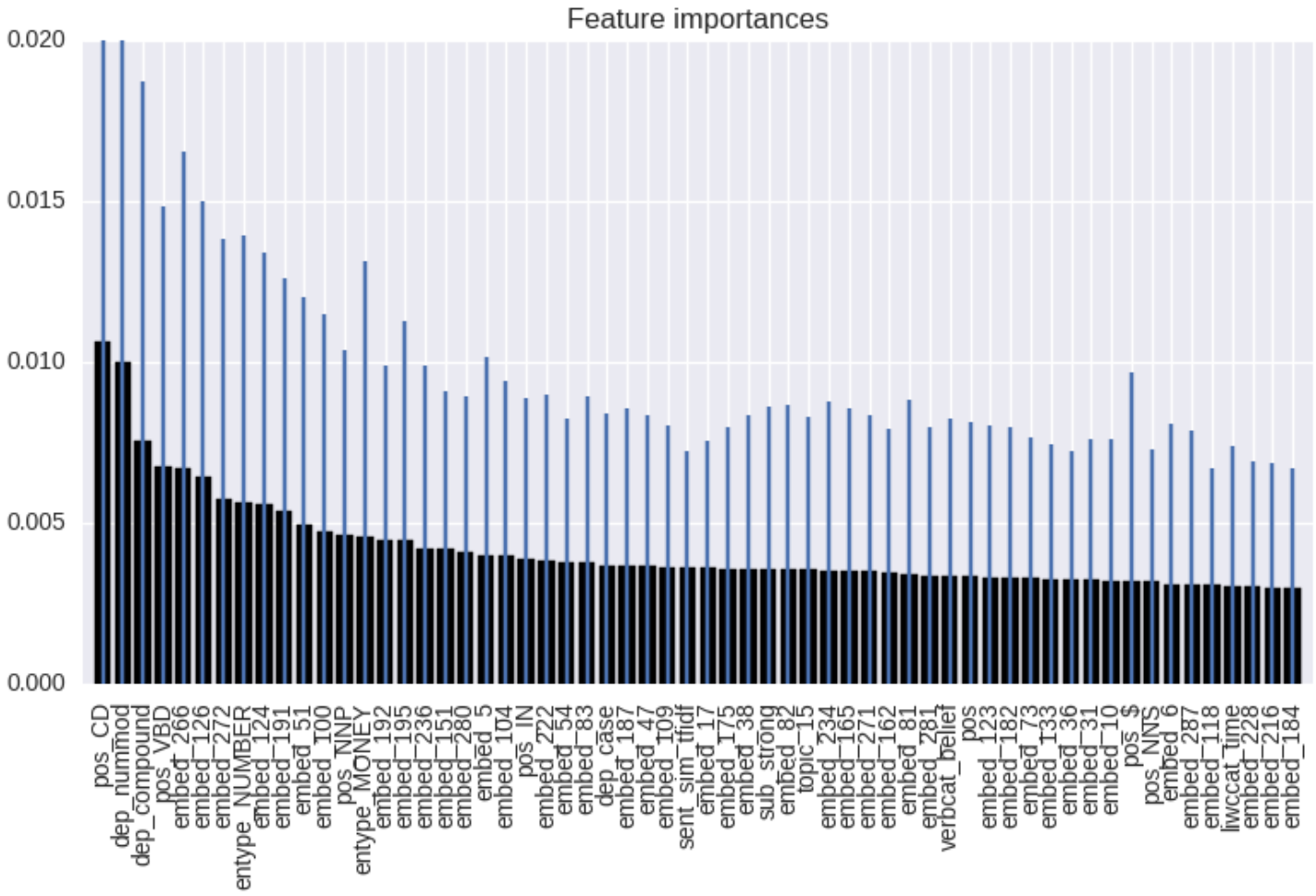


Figure 3.3: Feature Importance

3.4 Classification and Results

The dataset is divided into two sets: train set (70%) and test set (30%). Hence the train set consists of 426 sentences labeled as check-worthy and the test set consists of 182 sentences labeled as check-worthy.

Since the dataset is highly unbalanced (only 4.2% sentences are labeled check-worthy) we used under-sampling to balance the dataset. We randomly pick 426 samples from the non-labeled sentences in the train set and create a balanced dataset for training. We also under-sampled the test set similarly. We experimented with different classifiers, and the table 3.1 shows the performance of top classifiers.

We also use exploratory under-sampling (EasyEnsemble), i.e., create subsets of majority class and learning a classifier for each subset generated [24]. The performance of the best classifiers with different combinations of features is shown in 3.1. We also considered combining predictions of multiple classifiers trained separately and observe that it improves performance.

To evaluate the ranking performance of the system, we used the classifiers to rank the sentences in the test set in order of their check-worthiness, and measure the accuracy of top-k sentences using precision (P@k), average precision (AvgP) @k, and normalized discounted cumulative gain(nDCG) @k. The results are shown in figure 3.4 for best performing models.

As noted in section 3.1.1, the precision is low when tested on the natural test set because there is around 10% error in the labels of the non-checkworthy sentences.

Feature Combination	Precision	Recall	F1 Score
All features	0.1	0.75	0.18
Entity type	0.1	0.38	0.16
LIWC	0.06	0.57	0.11
Topics	0.06	0.65	0.11
Sentence Homogenity	0.06	0.54	0.11
Bigrams	0.09	0.4	0.15
Embeddings	0.178	0.309	0.224
POS Tags	0.1	0.63	0.17
POS+ET+HOM+B+Sm+S+LIWC	0.1	0.63	0.17
POS+ET+HOM+Sm+S	0.1	0.64	0.17
POS+ET+HOM+Sm+S+D	0.11	0.65	0.19
POS+ET+HOM+B+E	0.11	0.7	0.19
POS+TFIDF	0.09	0.64	0.16
TOP150(out of all leaving tfidf)	0.1	0.71	0.18

Table 3.1: Performance of best classifiers with different combinations of features

Dataset	**Threshold**	**Precision**	**Recall**	**F-1 Score**
Test Set	>0.45	0.17	0.37	0.23
	>0.499	**0.18**	**0.30**	**0.23**
	>0.5	0.18	0.27	0.22
	>0.6	0.20	0.12	0.15
Whole Dataset	>= 0.5	0.19	0.30	0.23

Table 3.2: Classification Performance of ClaimBuster

3.5 ClaimBuster Performance

We evaluate the performance of ClaimBuster[10] [44] on our dataset to compare our system with it. ClaimBuster returns check-worthiness scores for sentences through the web interface. Hence to obtain classification accuracy we use different thresholds on the score to classify as positive/negative. The results are shown in table 3.2

[10]idir-server2.uta.edu/claimbuster

Model	Precision	Recall	F1-Score
ClaimBuster	**0.18**	**0.30**	**0.23**
Embed_SVM+EnsRF	**0.178+/-0.012**	**0.309+/-0.016**	**0.224+/-0.012**
Embed_SVM	0.15	0.35	0.21
Embed_EnsRF	0.10	0.74	0.18
All_SVM	0.10	0.75	0.18
All_EnsRF	0.11	0.74	0.19

Table 3.3: Classification Performance: Comparison

We also investigate ClaimBuster's performance in ranking the sentences according to its check-worthiness. Figure 3.4 shows the ranking performance of ClaimBuster on the test set in terms of precision@k, average precision@k and NDCG@k.

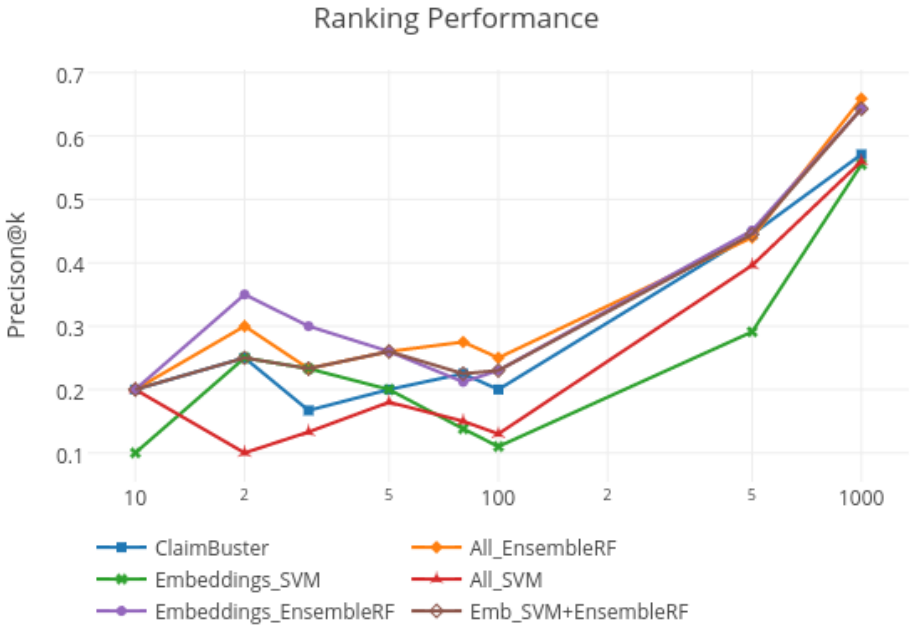
3.6 Comparison

Table 3.3 shows the precision, recall and f1-score of our good performing systems as compared with that of ClaimBuster on the test set. The results for ClaimBuster here correspond to a threshold of 0.5 which was their best performance on the test set.

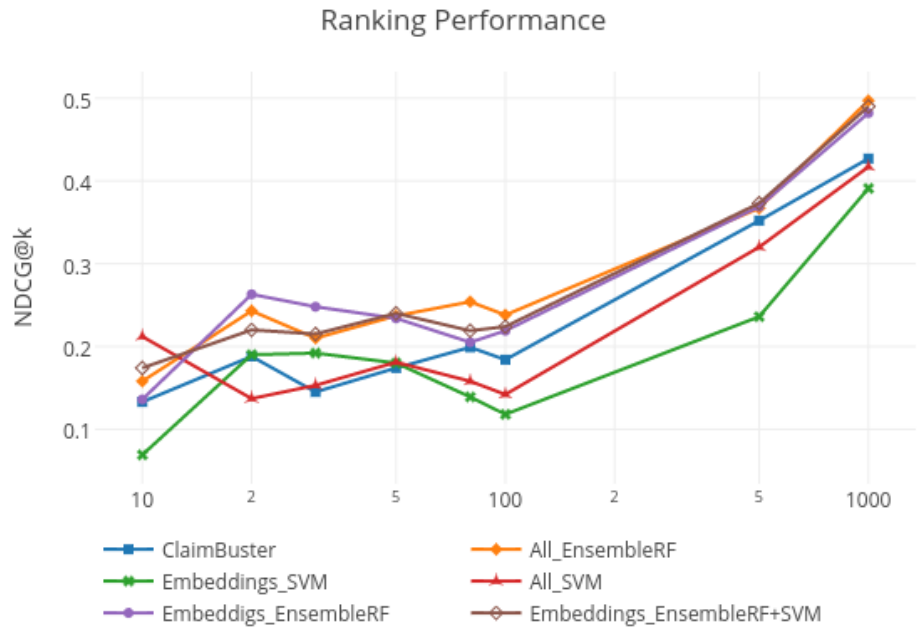
From the table we can see that our model achieves comparable performance to ClaimBuster in classification. Linear models like SVM are much better than ClaimBuster as far as recall is considered (0.74 as compared to 0.30) but ensemble models have a better precision at the cost of recall.

Though we achieve only comparable performance in classification, our models are much better than that of ClaimBuster in ranking. Figures 3.4a, 3.4c, 3.4b shows the precison@k, average precision@k and NDCG@k for different models.

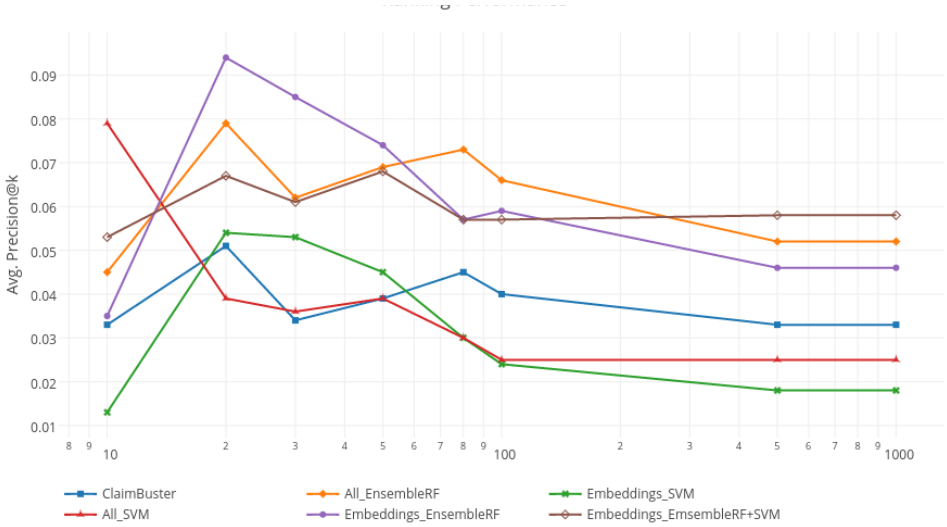
Since ranking is more important in this task because it is much more useful to have the most check-worthy sentences at top ranks so that down-stream



(a) Precision@k for different models



(b) NDCG@k for different models



(c) Average precision@k for different models

Figure 3.4: Ranking Performance of different models

systems or journalists can establish truthfulness of the sentence/claim.

Chapter 4

Stance Detection

Stance detection is an important component of fact-checking. A fact-checking system should be able to understand what how different information sources available over the web are related to each other and to the fact in question. It is often required to estimate the relative perspective of two given pieces of texts to extract related sources, corroborate different related sources, and finally use them in establishing the veracity of the fact.

We have explored solutions to this requirement of fact- checking system, namely the stance detection problem. We describe our system in the next sections. Section 4.1 describes the dataset and the fake news challenge which released the dataset. Section 4.2 describes a strong baseline system, which is the official baseline for the task. Section 4.3 discusses our approach to view the task as two sub-problems and sections 4.4 and 4.5 then describes our system and results for each subproblem respectively. Section 4.6 draws a comparison between the official baseline and our best performing system. We explore different neural net architectures for stance detection in news articles.

4.1 Dataset

We have used the FNC Stance detection dataset provided by the Fake News Challenge. The data is derived from the Emergent Dataset created by Craig Silverman [35, 12].

FNC (Fake News Challenge [1]) is a competition to encourage the exploration of AI (Artificial Intelligence) and NLP (Natural Language Processing) in combating the problem of fake news. As a stepping stone to the bigger goal, the challenge's first task is Stance Detection.

The stance detection task introduced by FNC1 aims to identify perspectives of news body texts toward headlines. Specifically, given a pair of headline and news article text, the goal is to predict whether the article text is related to the headline, and what the exact relationship is between them. Four different labels can be assigned to each headline-article pair: Unrelated, Discuss, Agree and Disagree. The distribution of the different labels in the dataset is shown in 4.1. Typically, a headline can be one sentence or a combination of several short expressions, and an article consists of several or even tens of sentences. An example is provided in Figure 4.2.

The FNC stance detection training dataset consists of two files, one with news article body IDs and content and the other with body IDs, stances and corresponding labels for the stance-body pairs. Only training data has been released at the time of this project. It consists of 1648 distinct headlines, 1683 distinct articles, and 49972 distinct headline-article pairings. Special care has to be taken for splitting the datasets as random splitting of dataset would lead to bleeding. The dataset has been split into three sets, train, development and test, ensuring that the three sets have no overlapping article bodies. The size of the sets are given in Table 4.1.

[1] http://www.fakenewschallenge.org/

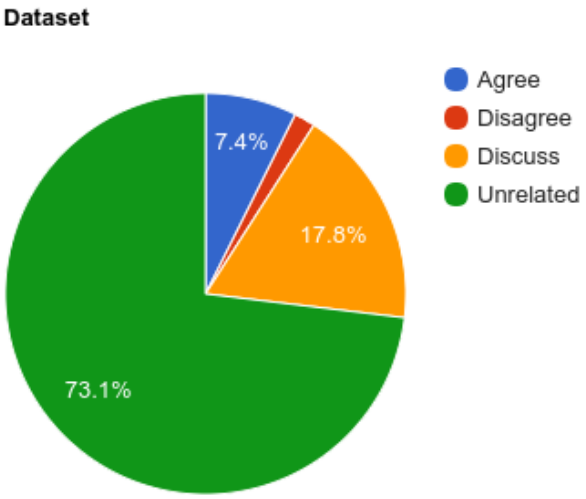


Figure 4.1: Distribution of gold labels for headline-body pairs in the fnc dataset

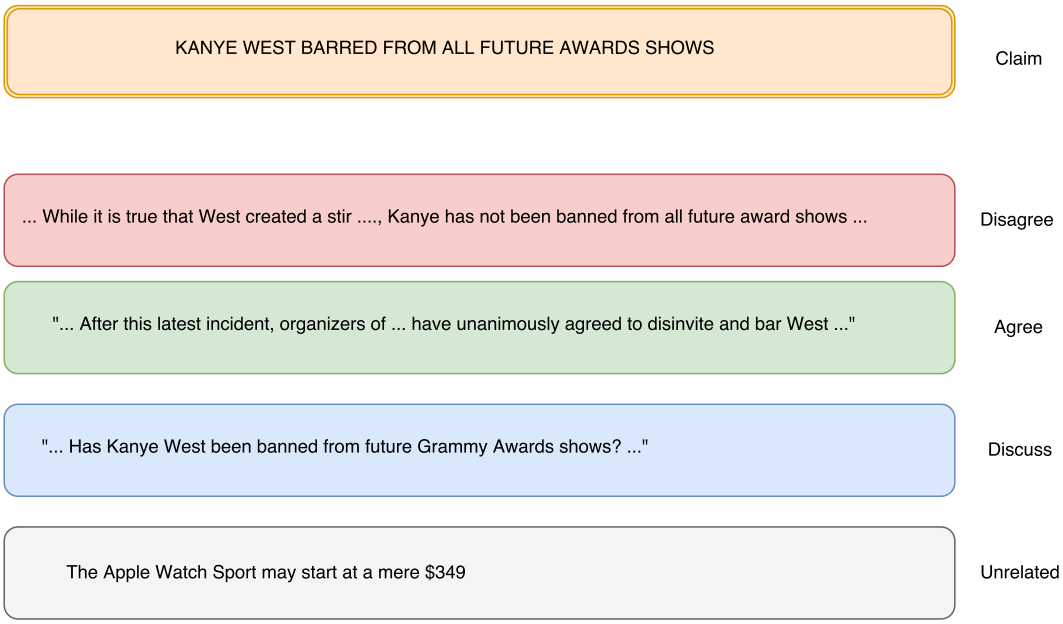


Figure 4.2: An Example

Data Partition	#Total Pairs	#Related Pairs	Percentage
Train	32400	8613	64.8%
Development	7950	2090	15.9%
Test	9622	2724	19.3%

Table 4.1: The Dataset Split

4.2 Baseline

A simple baseline using hand-coded features and a GradientBoosting classifier has been provided by the fnc organizers.

Some of the hand-crafted features used in the baseline are:-

- **Binary co-occurrence**: The number of times a token in the headline appears in the body text.
- **Count n-grams**: The number of times an n-gram of the headline appears in the text. Different n-grams namely character 2-grams, 4-grams, 8-grams, 16-grams and word 2-grams, 3-grams, 4-grams, 8-grams and 16-grams, are considered.
- **Refuting words**: For each refuting word, a binary value is added to the features which is 1 if the refuting word is present in the headline/body and 0 if it is not. Refuting words include words like 'fake', 'fraud', 'hoax', 'false', 'deny', 'denies', 'not' etc.
- **Polarity**: It is 1 if number of refuting words is odd else 0.
- **Word Overlap**: It is equal to the number of distinct common words in headline and body divided by the total number of distinct words in them.

With these features and a gradient boosting classifier, the baseline achieves

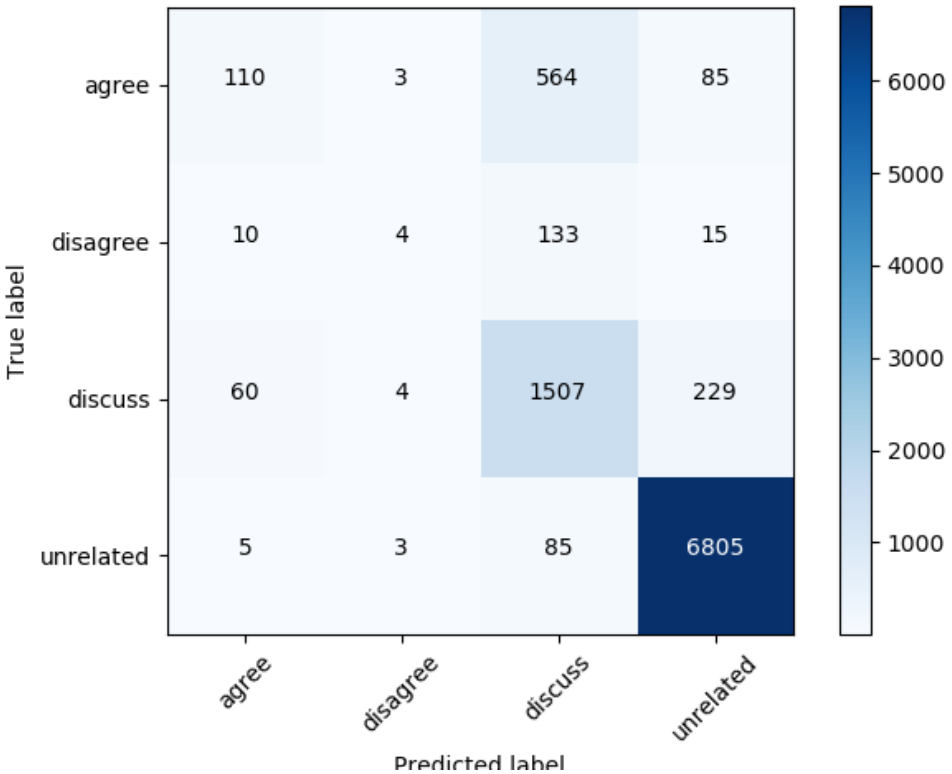


Figure 4.3: Confusion Matrix for Baseline

a weighted accuracy score of 79.03% on the test set. The confusion matrix using the baseline is presented in figure 4.3.

4.3 Approach

The task of classifying a news article relative to a news headline as agreeing, discussing (without taking a position), disagreeing, or unrelated is composed of two related yet different sub-problems:

1. classifying whether the news article-headline pair is related or unrelated, and

2. classifying the type of stance of a news article with respect to the headline, if related

Also, in the dataset, 73% of the samples are unrelated, while the other classes combined constitute only the rest 27%. Hence we approach the task as two classification problems so that the models for each sub-problem can capture the relevant specificities of the problem better and also in the process do away with class imbalance problems.

4.4 Subproblem 1: Related-Unrelated classification

The sub-problem is a simple classification problem to classify if two text pieces are related.

4.4.1 Features

We explored different combinations of the following features:

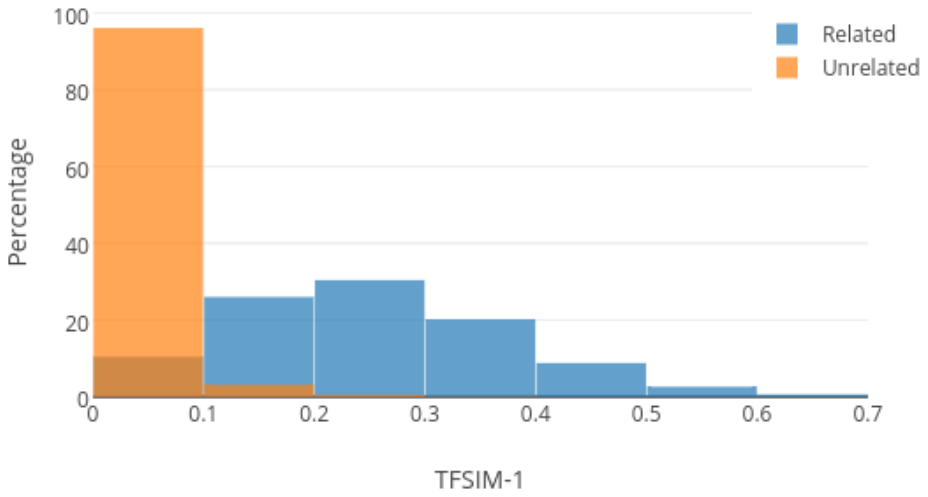


Figure 4.4: Distribution of dataset with tf-idf similarity (TFSIM-1)

- **TF-IDF Similarity**: The cosine similarity between the tf-idf vectors of the headline and the news article is taken as a feature. We considered two tf-idf models: one using only unigrams (TFSIM-1) and the other using both unigrams and bigrams (TFSIM-2). In both cases, types occurring in more than 95% documents and less than 5 documents were removed. Figure 4.4 shows the distribution of the samples in the dataset with tf-idf similarity (TFSIM-1).

- **Named Entity Overlap**: The number of named entities common between the headline and the news article is taken as a feature. The named entities were extracted using Stanford's NER tagger [13]. Two variations differing in the types of named entities considered were developed:

  1. *CNTNE-1*: All types of named entities were considered.
  2. *CNTNE-2*: Only named entities of classes PERSON, LOCATION,

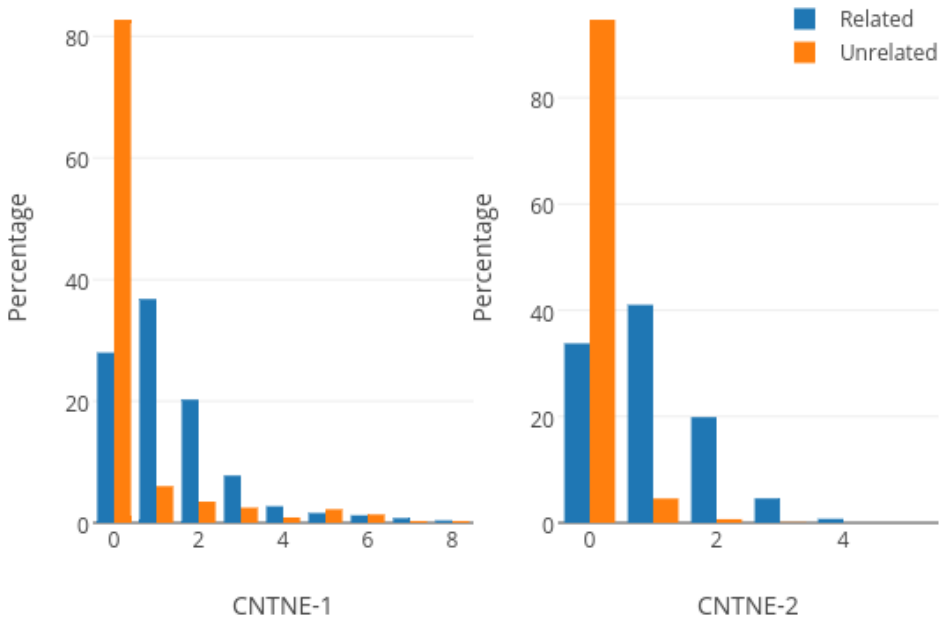


Figure 4.5: Distribution of the samples in the dataset with CNTNE-1 and CNTNE-2

ORGANIZATION, MONEY, PERCENT were considered.

Figure 4.5 shows the distribution of the samples in the dataset with CNTNE-1 and CNTNE-2.

- **Word Overlap**: Number of tokens in headline present in the first 100 tokens of the article (OVLP).

4.4.2 Classification and Results

We considered different combinations of features described. For each combination, different classifiers such as SVM, Naive Bayes, Logistic Regression and Gradient boosting were trained on the train set and the classifier with the best accuracy on the validation set was chosen. The performance of the classifiers on the test set for different combinations of the features is shown in table 4.2.

Features	Precison	Recall	F1-Score	Accuracy
TFSIM-1	0.92	0.86	0.89	91.43%
CNTNE-2	0.88	0.81	0.83	87.52%
TFSIM1+ CNTNE2	0.95	0.93	0.94	94.98%
TFSIM2+ CNTNE2	0.94	0.92	0.93	94.12%
TFSIM2+ OVLP+ CNTNE2	**0.96**	**0.95**	**0.95**	**96.35%**
Baseline OB	0.96	0.93	0.94	95.61%

Table 4.2: Performance of different combination of features on test set (Precision, Recall and F1-Score are macro-averaged)

Hence we see that our model with only 3 simple features performs better than the official baseline which uses so many features and hence is much slower.

4.5 Subproblem 2: Stance classification

Given related pairs of headline and articles, the task in this subproblem is to classify them as agreeing, disagreeing or discussing.

4.5.1 Models

We used various neural network approaches for the task. In order to transform the input words into vector space, we used 300 Dimensional GloVe vectors trained on the 6B token set of Wikipedia and Common Crawl [32]. We further created a randomly initialized OOV vector for words that are not found in the Glove vocabulary. Since the article length is large, we take only the first 150 tokens of the article in all LSTM models. We discuss the architectures one by one below in detail:

MLP on doc2vec sentence embeddings (MLPDoc2Vec)

We used a multilayer perceptron with 2 hidden layers and ReLU activation to classify the examples. The input were the doc2vec embeddings for the two sentences concatenated and the cross entropy with softmax activation function was used at the last layer. The input layer was of size 600, first hidden layer of size 100, second hidden layer of size 10 and the output layer of size 3. Since the dataset was very unbalanced, we also trained the model on balanced but very small dataset where there are 1533 examples of "agree" and "discuss" class and 511 examples of "disagree" class (MLPDoc2VecTrunc)

Convolutional Neural Network (CNNGlove)

In this model [22], we made an image of each sentence by using the glove embeddings of the words in the sentence as the rows and columns as the different words of the sentence. The sentences were padded to length 95 by using zero vectors. Hence each sentence was represented as a 95X300 image. We performed one dimensional convolution on this image with 128 filters of size 3X300. Now we max pool the activations of each filter along the size of the image, thus giving us 128 features for each sentence. These 128 features are then concatenated for both the sentences to give us a 256 sized vector passed through a fully connected layer to a hidden layer of 16 neurons, which is fully connected to the 3-sized output layer. ReLU is used at all the hidden layers. We use a dropout with keep probability of 0.95 at the last hidden layer and softmax classifier at the output layer with cross entropy as the loss measure. The model is depicted in Figure 4.6.

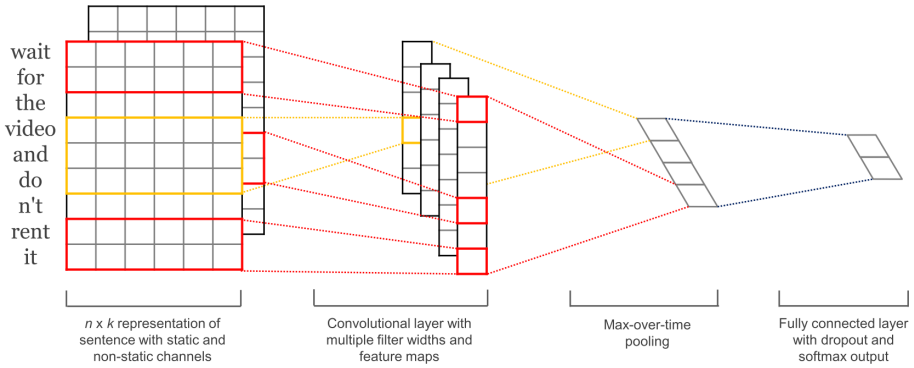


Figure 4.6: CNN model for sentence embedding

Conditional LSTM Encodings (CondLSTM)

Using a single LSTM cell to encode the concatenated headline-article pair fails to capture the dependence and relation between the two. Hence we use conditional LSTM encoders. In this architecture, we use two separate LSTMs, one for the headline and one for the article. The first LSTM scans the headline sequence to encode the headline as h. Now the hidden state second LSTM cell is initialized with h and thus the article is encoded "conditional" to the headline. The final output of the second LSTM is passed through a fully connected layer to get the classification layer with softmax activation and trained with cross entropy loss.

Bidirectional Conditional Encoding (BiDiCond)

This is the same as the above model for conditional encoding with just one change that we move both forwards and backwards in time. The final encoding is the concatenation of the final hidden states of the LSTMs in the two directions. The model is explained graphically in the figure 4.7. [1]

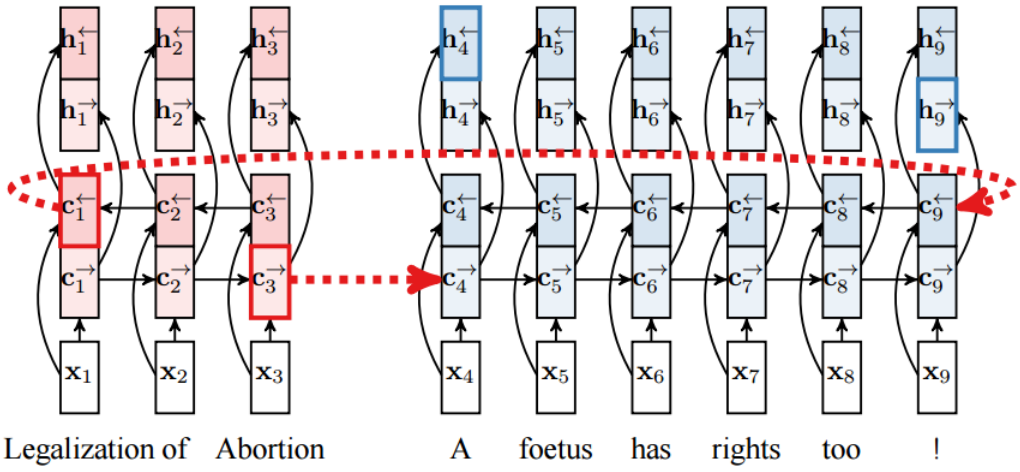


Figure 4.7: Bidirectional Conditional model

Conditional Encoding With Attention

Inspired by the success of attention in seq2seq models [2] we use the attention in the conditional encoding [34](4.8) to further improve performance. We consider the following variations:

1. Conditional Encoding with global attention (CondGlobal).
2. Conditional Encoding with word-by-word attention (CondWord).
3. Bidirectional Conditional Encoding with word-by-word attention (BiDiWord).
4. Bidirectional Conditional Encoding with Global attention (BiDiGlobal).

Sentence Representation using SPINN (Spinn)

We use the SPINN model [4] to obtain a rich representation of a sentence which we then use for stance classification. The SPINN model combines parsing and interpretation within a single tree-sequence hybrid model by integrating tree-

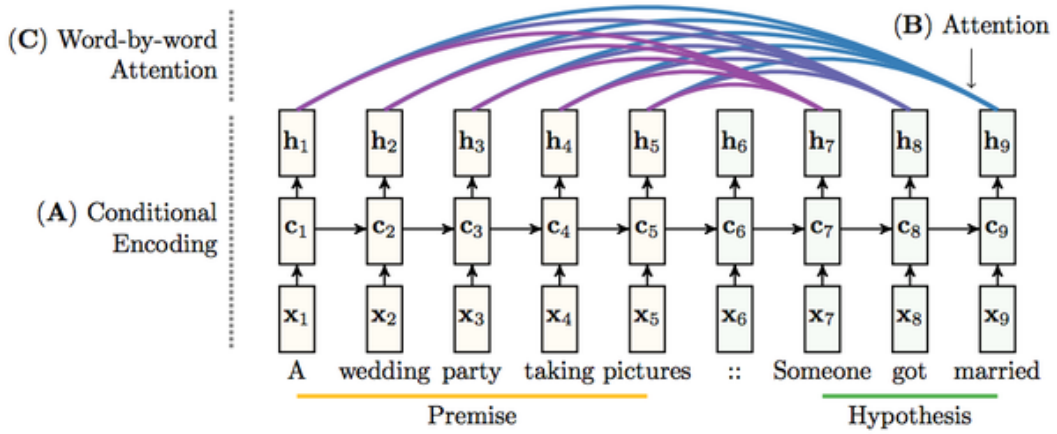


Figure 4.8: Global and Word-by-Word Attention

structured sentence interpretation into the linear sequential structure of a shift reduce parser.

To classify a headline-article pair, we run two copies of SPINN with shared parameters: one on the headline sentence and another on the article sentence. We use only the first sentence of the news article in this model as it is quite representative of the article. We then use their outputs to construct a feature vector. The feature vector is formed from the concatenation of the two vectors, their difference and their element-wise product.

This feature vector is then passed through a 1-layer(1024D) MLP, and finally passed through a softmax layer, to obtain a distribution over the three labels. The model is then trained to minimize the cross-entropy loss.

4.5.2 Results

The performance of difference models is shown in table 4.3. The macro-averaged precision, recall and f1-score is shown in the table. We can observe that most deep models outperform the baseline. However many models without attention fail to predict any headline-article pair in the "disagree" class because they are too simple to capture the nuances of the task. Even Cond-

Features	Precison	Recall	F1-Score	Accuracy
CNNGlove	0.22	0.33	0.26	66.07%
MLPDoc2vec	0.41	0.40	0.39	68.61%
MLPDoc2vecTrunc	0.51	0.45	0.40	56.68%
CondLSTM	0.42	0.42	0.41	68.80%
BiDiCond	0.44	0.46	0.45	71.18%
CondGlobal	0.44	0.47	0.45	70.96%
BiDiGlobal	0.61	0.56	0.57	74.08%
CondWord	0.50	0.49	0.48	70.81%
BiDiWord	0.64	0.58	0.59	74.52%
Spinn	0.59	0.56	0.57	74.08%
Baseline OB	0.55	0.38	0.36	67.68%

Table 4.3: Performance of different models including the baseline

Global fails to predict any pair from the disagree class. The bidirectional models perform well on the dataset with the word-by-word attention model outperforming all others with an accuracy of 74.52%. The model based on SPINN also performs w with an accuracy of 74.08%.

4.6 Comparison

Fake News challenge defines a score to evaluate the performance of a system on their dataset. The system is evaluated based on a weighted, two level scoring system. Level 1: Classify headline and body text as related or unrelated 25% score weighting Level 2: Classify related pairs as agrees, disagrees, or discusses 75% score weighting The rationale for this score is that the related/unrelated classification task is expected to be much easier and is less relevant for detecting fake news, so it should be given less weight in the evaluation metric. The Stance Detection task (classify as agrees, disagrees or discuss) is both more difficult and more relevant to fake news detection, so should be given much more weight in the evaluation metric. A schematic for the scoring is given in figure 4.9

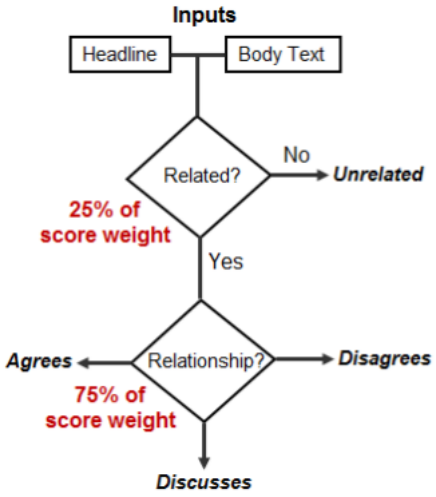


Figure 4.9: Scoring Process Schematic

Concretely, if a [headline, article] pair in the test set has the target label unrelated, the evaluation score will be incremented by 0.25 if it labels the pair as unrelated.

If the [headline, article] test pair is related, the score will be incremented by 0.25 if it labels the pair as any of the three classes: agrees, disagrees, or discusses.

The evaluation score will also be incremented by an additional 0.75 for each related pair if gets the relationship right by labeling the pair with the single correct class: agrees, disagrees, or discusses.

The official baseline approach gives a score of 79.03% (Score of 3515.75 out of best possible score 4448.5) for our test set while our system gives a score of 79.09% for BiDirGlobal model and 79.60% for BiDirWord model.

The confusion matrix for BiDirWord model is shown in figure 4.10

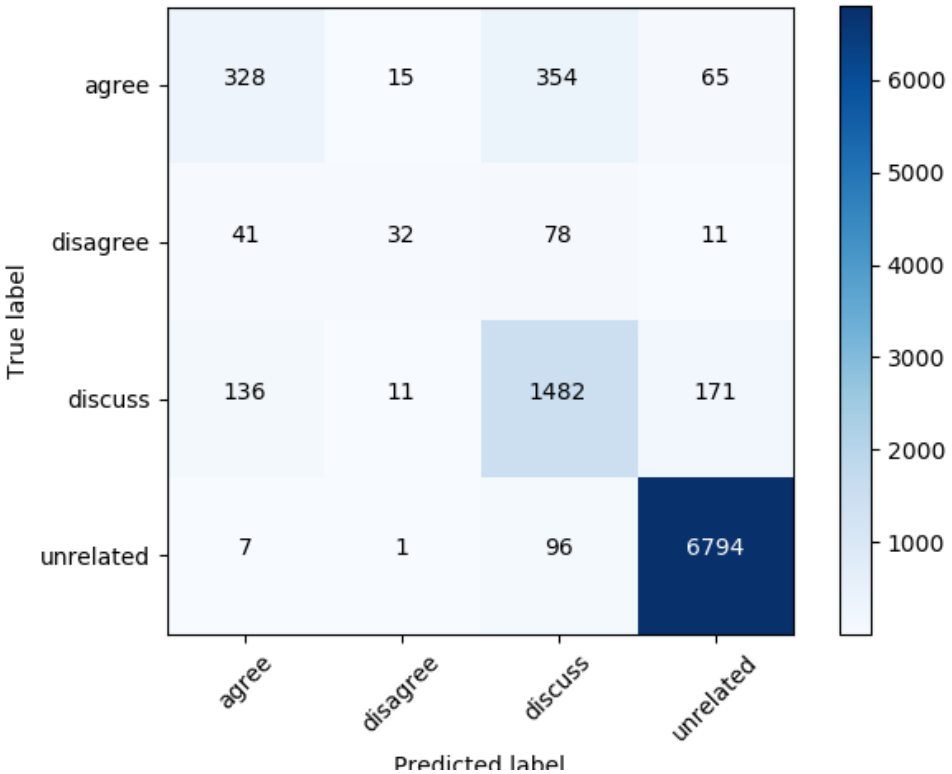


Figure 4.10: Confusion Matrix for BiDirGlobal model

Chapter 5

Future Work

For the task of extracting check-worthy claims, it is important that the system generalizes not only on the debates but also on interviews, speeches and on other forms of public interactions of the politicians. Another major challenge would be to build a system that does not go stale after a few years. Hence, there's need to update the world knowledge of the system regularly and this process might be automated too.

Currently, Stance detection systems are meant to allow a human fact checker to enter a claim or headline and instantly retrieve the top articles that agree, disagree or discuss the claim/headline in question. They could then look at the arguments for and against the claim, and use their human judgment and reasoning skills to assess the validity of the claim in question. Such a tool would enable human fact checkers to be fast and effective.

But the application of a stance detection solution would not be limited to that. It should be possible to build a prototype post-facto "truth labeling" system from a "stance detection" system. Such a system would tentatively label a claim or story as true/false based on the stances taken by various news organizations on the topic, weighted by their credibility. In this way, the various stances (or lack of a stance) news organizations take on a claim, as

determined by an automatic stance detection system, could be combined to tentatively label the claim as True or False. While crude, this type of fully-automated approach to truth labeling could serve as a starting point for human fact checkers, e.g. to prioritize which claims are worth further investigation. This could be a possible direction for future works.

In the current times or in the near future, well-defined narrow scoped statements like "US Unemployment went up during the Obama years" could be fact checked automatically with a reasonably amount of additional research. But a statement like: "The Russians under Putin interfered with the US Presidential Election" wont be possible to fact check automatically until we've achieved human-level artificial intelligence capable of understanding subtle and complex human interactions, and conducting investigative journalism. That's why it is very important to break down the problem of automated fact checking into very small achievable targets which may serve as a useful tool for human fact-checkers if used today!

Even though we might not reach the "Holy Grail" in near future, it would be great if we regularly provide journalists with new assisting tools that make their work easier, faster and more accurate.

Bibliography

[1] Isabelle Augenstein et al. "Stance detection with bidirectional conditional encoding". In: *arXiv preprint arXiv:1606.05464* (2016).

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[3] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation". In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[4] Samuel R Bowman et al. "A fast unified model for parsing and sentence understanding". In: *arXiv preprint arXiv:1603.06021* (2016).

[5] Samuel R Bowman et al. "A large annotated corpus for learning natural language inference". In: *arXiv preprint arXiv:1508.05326* (2015).

[6] Danqi Chen, Jason Bolton, and Christopher D Manning. "A thorough examination of the cnn/daily mail reading comprehension task". In: *arXiv preprint arXiv:1606.02858* (2016).

[7] Giovanni Luca Ciampaglia et al. "Computational fact checking from knowledge networks". In: *PloS one* 10.6 (2015), e0128193.

[8] Sarah Cohen, James T Hamilton, and Fred Turner. "Computational journalism". In: *Communications of the ACM* 54.10 (2011), pp. 66–71.

[9] Sarah Cohen et al. "Computational Journalism: A Call to Arms to Database Researchers." In: *CIDR*. Vol. 2011. 2011, pp. 148–151.

[10] Katy Davis. "Study: Fact-checkers disagree on who lies most". In: *Center for Media and Public Affairs* (2012).

[11] Graciana Diez-Roux et al. "A high-resolution anatomical atlas of the transcriptome in the mouse embryo". In: *PLoS biology* 9.1 (2011), e1000582.

[12] William Ferreira and Andreas Vlachos. "Emergent: a novel data-set for stance classification". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. ACL. 2016.

[13] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. "Incorporating non-local information into information extraction systems by gibbs sampling". In: *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics. 2005, pp. 363–370.

[14] Terry Flew et al. "The promise of computational journalism". In: *Journalism Practice* 6.2 (2012), pp. 157–171.

[15] Eibe Frank and Mark Hall. "A simple approach to ordinal classification". In: *European Conference on Machine Learning*. Springer. 2001, pp. 145–156.

[16] Lucas Graves. *Deciding what's true: Fact-checking journalism and the new ecology of news*. Columbia University, 2013.

[17] Kazi Saidul Hasan and Vincent Ng. "Stance Classification of Ideological Debates: Data, Models, Features, and Constraints." In: *IJCNLP*. 2013, pp. 1348–1356.

[18] Naeemul Hassan, Chengkai Li, and Mark Tremayne. "Detecting check-worthy factual claims in presidential debates". In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM. 2015, pp. 1835–1838.

[19] Naeemul Hassan et al. "The Quest to Automate Fact-Checking". In: *world* (2015).

[20] Karl Moritz Hermann et al. "Teaching machines to read and comprehend". In: *Advances in Neural Information Processing Systems*. 2015, pp. 1693–1701.

[21] Clayton J Hutto and Eric Gilbert. "Vader: A parsimonious rule-based model for sentiment analysis of social media text". In: *Eighth International AAAI Conference on Weblogs and Social Media*. 2014.

[22] Yoon Kim. "Convolutional neural networks for sentence classification". In: *arXiv preprint arXiv:1408.5882* (2014).

[23] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. "Meme-tracking and the dynamics of the news cycle". In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2009, pp. 497–506.

[24] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. "Exploratory undersampling for class-imbalance learning". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.2 (2009), pp. 539–550.

[25] Christopher D. Manning et al. "The Stanford CoreNLP Natural Language Processing Toolkit". In: *Association for Computational Linguistics (ACL) System Demonstrations*. 2014, pp. 55–60. URL: `http://www.aclweb.org/anthology/P/P14/P14-5010`.

[26] Saif M Mohammad et al. "Semeval-2016 task 6: Detecting stance in tweets". In: *Proceedings of SemEval* 16 (2016).

[27] Akiko Murakami and Rudy Raymond. "Support or oppose?: classifying positions in online debates from reply activities and opinion expressions". In: *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. Association for Computational Linguistics. 2010, pp. 869–875.

[28] Ndapandula Nakashole and Tom M Mitchell. "Language-Aware Truth Assessment of Fact Candidates." In: *ACL (1)*. 2014, pp. 1009–1019.

[29] Brendan Nyhan and Jason Reifler. "The Effect of Fact-checking on Elites". In: (2014). URL: `http://thedata.harvard.`.

[30] James W Pennebaker, Martha E Francis, and Roger J Booth. "Linguistic inquiry and word count: LIWC 2001". In: *Mahway: Lawrence Erlbaum Associates* 71.2001 (2001), p. 2001.

[31] James W Pennebaker et al. *The development and psychometric properties of LIWC2015*. Tech. rep. 2015.

[32] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543. URL: `http://www.aclweb.org/anthology/D14-1162`.

[33] Xuan-Hieu Phan and Cam-Tu Nguyen. "GibbsLDA++: A C/C++ implementation of latent Dirichlet allocation (LDA)". In: (2007).

[34] Tim Rocktäschel et al. "Reasoning about entailment with neural attention". In: *arXiv preprint arXiv:1509.06664* (2015).

[35] Craig Silverman. "Lies, Damn lies, and viral content. How news websites spread (and debunk) online rumors, unverified claims, and misinformation". In: *Tow Center for Digital Journalism* (2015).

[36] Swapna Somasundaran and Janyce Wiebe. "Recognizing stances in online debates". In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics. 2009, pp. 226–234.

[37] Dhanya Sridhar, Lise Getoor, and Marilyn Walker. "Collective stance classification of posts in online debate forums". In: *ACL 2014* 109 (2014).

[38] Matt Thomas, Bo Pang, and Lillian Lee. "Get out the vote: Determining support or opposition from Congressional floor-debate transcripts". In: *Proceedings of the 2006 conference on empirical methods in natural language processing*. Association for Computational Linguistics. 2006, pp. 327–335.

[39] Andreas Vlachos and Sebastian Riedel. "Fact Checking: Task definition and dataset construction". In: *ACL 2014* (2014), p. 18.

[40] Marilyn A Walker et al. "Stance classification using dialogic properties of persuasion". In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics. 2012, pp. 592–596.

[41] Zhiguo Wang, Wael Hamza, and Radu Florian. "Bilateral Multi-Perspective Matching for Natural Language Sentences". In: *arXiv preprint arXiv:1702.03814* (2017).

[42] Janyce Wiebe and Rada Mihalcea. "Word sense and subjectivity". In: *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics. 2006, pp. 1065–1072.

[43] Thomas Wood and Ethan Porter. "The Elusive Backfire Effect: Mass Attitudes' Steadfast Factual Adherence". In: *Available at SSRN 2819073* (2016).

[44] You Wu et al. "Toward computational fact-checking". In: *Proceedings of the VLDB Endowment* 7.7 (2014), pp. 589–600.