Detecting Check-Worthy Claims

by

Oliver Tianhao Ren

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2019

Author . . . . . .
Department of Electrical Engineering and Computer Science
June 7, 2019

Certified by . . . . . .
James R. Glass
Senior Research Scientist
Thesis Supervisor

Certified by . . . . . .
Mitra Mohtarami
Researh Scientist
Thesis Supervisor

Accepted by . . . . . .
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Detecting Check-Worthy Claims

by

Oliver Tianhao Ren

Submitted to the Department of Electrical Engineering and Computer Science
on June 7, 2019, in partial fulfillment of the
requirements for the degree of
Master of Computer Science and Engineering

Abstract

An automated fact-checking system aims to check the factuality of published information, such as news articles or blog posts. It is a time-consuming process and contains several steps: (*i*) detecting check-worthy claims, (*ii*) extracting a set of evidence for a given worthy claim from reliable sources, (*iii*) predicting the stances of the set of evidence with respect to the claim, (*iv*) integrating all information from (*ii*) and (*iii*) to detect whether the claim is factually true or false. In this thesis, we focus on the first step that is detecting check-worthy claims. It aims to facilitate manual fact-checking efforts by prioritizing the claims that fact-checkers should consider first.

Previous works [Jaradat et al., 2018; Gencheva et al., 2017] for detecting check-worthy claims mainly focused on a debate domain, and used a set of hand-crafted features and simple semantic representations at the sentence level based on averaged `word2vec` embeddings [Mikolov et al., 2013]. Their limitations are (*i*) less attention to the context and text representation, and (*ii*) the limited size of the developed debate dataset that is not enough to train the neural network models. To address these limitations, in this thesis, we investigate different approaches to incorporate context both at word and sentence levels using recurrent neural networks (RNNs), Embeddings from Language Models (ELMo) [Peters et al., 2018] and Bidirectional Encoder Representations from Transformers (BERT) [Devlin et al., 2018]. We show our model based on BERT can outperform the state-of-the-art baseline on the presidential debate dataset [Gencheva et al., 2017]. In addition, we extend the Debate dataset and create a new dataset that uses Wikipedia inline citations as a proxy for check-worthiness. The dataset contains millions of check-worthy claims and covers various domains. Our experiments on this dataset show the useful features correlated with check-worthiness can vary across domains.

Thesis Supervisor: James R. Glass
Title: Senior Research Scientist

Thesis Supervisor: Mitra Mohtarami
Title: Researh Scientist

Acknowledgments

I would like to thank my thesis advisors Mitra Mohtarami and James Glass for all their help in shaping the direction of my research. Without their continued efforts, this thesis would not have been possible. I would like to thank the other members of the SLS Fake News group Ramy Baly, Wei Fang, and Moin Nadeem for assisting me with my research and being available to bounce ideas off of. I would like to thank Qatar Computing Research Institute (QCRI) for their support and funding my research. I would also like to thank my friends and family for supporting me through my research. I would like to give a special thank you to my MIT ACF family being with me and motivating me during all those late nights I spent running my experiments and writing my thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Modern technology reaches further than it did during any other time in history. At the touch of a screen, consumers can immediately access incredible amounts of information, all at the speed of light. However, the speed and ease at which information–credible or incredible–travels today comes with a cost. Readers are constantly inundated with new stories, making it difficult to differentiate between real and fake information. Being able to distinguish between the two is important because individuals make decisions based on the information that they have. A desire to make it easier and to make informed decisions motivates our pursuit of a more robust and automatic fact-checking system.

The input to an automatic fact-checking system would be an unannotated document, and the output would be a document annotated with *true* or *false* labels. The automatic fact-checking procedure can be split into several steps [Vlachos and Riedel, 2014; Mihaylova et al., 2018]. The first step is to identify the check-worthy claims in a document [Atanasova et al., 2018; Jaradat et al., 2018]. The second step is to retrieve evidence related to the identified claims from reliable sources. The third step is to identify the stances for the retrieved set of evidence with respect to the identified claims [Mohtarami et al., 2018; Inkpen et al., 2017; Kochkina et al., 2018; Dungs et al., 2018; Bar-Haim et al., 2017]. The final step is to use the retrieved set

of evidence and their stances to identify the claims as factually *true* or *false*.

There has been work on checking the factuality/credibility of a claim, of a news article, or of an information source [Thorne and Vlachos, 2018]. However, less attention has been paid to some steps of fact-checking like detection of the check-worthy claims which this thesis contributes to. In the next section, we discuss the specific research problem covered in this work.

1.2 Problem Description

This thesis focuses on the first step of an automatic fact-checking system: detecting check-worthy claims. A check-worthy claim is defined as one that a professional fact-checker would have actually fact-checked and one that would have an impact on individuals or social media. Currently, fact-checking is primarily done by professional fact-checkers employed by companies such as Snopes[1] or Politifact[2]. However, fact-checking is a painstaking and slow process, and most articles on Web have not been put through a fact-checker. As an independent system, our work would help to automate the process of manually deciding whether a statement is check-worthy or not through a document.

In this thesis, we aim to present a model that is able to assign check-worthy scores to the sentences of a given document. These scores can then be used to determine which sentences in the document are check-worthy and to rank the sentences in order of check-worthiness. A score is an approximation of the probability that the input sentence is check-worthy. That is, the sentences with higher scores are more likely to be check-worthy than the ones with lower scores.

One of the challenges of this task is the limited size of existing labeled data to be used for training the models. For instance, the CLEF-2018 claim-detection dataset has less than 9K claims [Atanasova et al., 2018]. There is another dataset based on journalist annotations of the 2016 Presidential election cycle [Gencheva et al., 2017].

[1] `https://www.snopes.com`
[2] `https://www.politifact.com`

However, it has some shortcomings and only a small number of claims are annotated as check-worthy, and it is not clear if the unannotated claims are really not-check-worthy or have just been left unannotated in the published articles. In addition, there are some fact-checking datasets such as the Fake News Challenge (FNC) [Pomerleau and Rao, 2017] and the Fact Extraction and VERification (FEVER) [Thorne et al., 2018] datasets, which do not include the required labels for a claim-detection task. Thus, we are not able to take advantage of those relatively much larger datasets.

In this thesis, we also address the challenge by creating a new dataset for the check-worthy claim detection task that is both much larger than existing datasets and covers different topics (the existing datasets for this task are excluded to the political domain). This allows us to have a comprehensive dataset of claims from outside the political sphere, which is the primary output of fact-checking organizations, to create more generalizable claim-detection models. Our presented dataset is based upon Wikipedia articles, because Wikipedia requires that all claims that might be challenged have a citation[3].

1.3 Contributions

Our direct contributions to the check-worthy claims detection task are as follows:

- We present various text representations and neural network models that are not based upon hand-crafted features, and our model can outperform the state-of-the-art performance [Jaradat et al., 2018] on a publicly available debate dataset.

- We show the differences in feature importance between check-worthy and not-check-worthy claims and between spoken and written claims.

- We extend the existing dataset in the political domain, and also develop a new dataset with various domains based upon Wikipedia articles that contains millions more claims than existing claim-detection datasets and is more complete with its labeling.

[3] `https://en.wikipedia.org/wiki/Wikipedia:Citing_sources`

- We then extend our models and experiments to be applied on our developed dataset.

Each of these contributions will be discussed in detail in the remainder of this thesis. The next section provides an outline of the following chapters.

1.4 Outline

In Chapter 2, we discuss previous work related to the fact-checking and claim-detection tasks. For the remaining chapters, we organize our findings based on the dataset used/developed for our experiments. Chapter 3 focuses on our work with the existing dataset [Gencheva et al., 2017; Atanasova et al., 2018] in the presidential debate domain. In Chapter 4, we present a new dataset in political domain based on the fact-checked articles released by Politifact[4]. We shift gears in Chapter 5 and focus on our experiments with another new dataset with various domains based upon Wikipedia and its internal methodology for determining check-worthy claims. In addition, we develop another new dataset for a competition on claim-detection task in Conference and Labs of the Evaluation Forum (CLEF)-2019 explained in Chapter 6. Finally, in Chapter 7, we summarize our overall findings and discuss possibilities for future work.

[4] `https://www.politifact.com`

Chapter 2

Related Work

2.1 Fact Checking

As mentioned in Chapter 1, an automated fact-checking system is a multi-step process: (*i*) detecting check-worthy claims, (*ii*) providing evidence, (*iii*) detecting stance of the evidence to the input claims, and finally (*iv*) checking the factuality of the claims.

The first step in fact-checking is detecting which claims in a document are even worth fact-checking. One currently available model that predicts check-worthiness is the ClaimRank model [Gencheva et al., 2017; Jaradat et al., 2018]. The ClaimRank [1] model was trained on a set of manually-annotated claims, extracted from the 2016 US presidential and vice-presidential debates, which were gathered from nine reputable sources. The model used both the textual context and the overall context of the claim within the debate in order to make its predictions. The model is currently online and an example output can be seen in Figure 2-1. The same dataset was provided as part of the CLEF-2018 challenge, and the most successful models in the challenge showed the potential effectiveness of recurrent neural network based approaches [Atanasova et al., 2018].

The second step of the fact-checking process is finding relevant external resources related to a input claim. The factuality of the claim is determined by its relationship

[1] `http://claimrank.qcri.org`

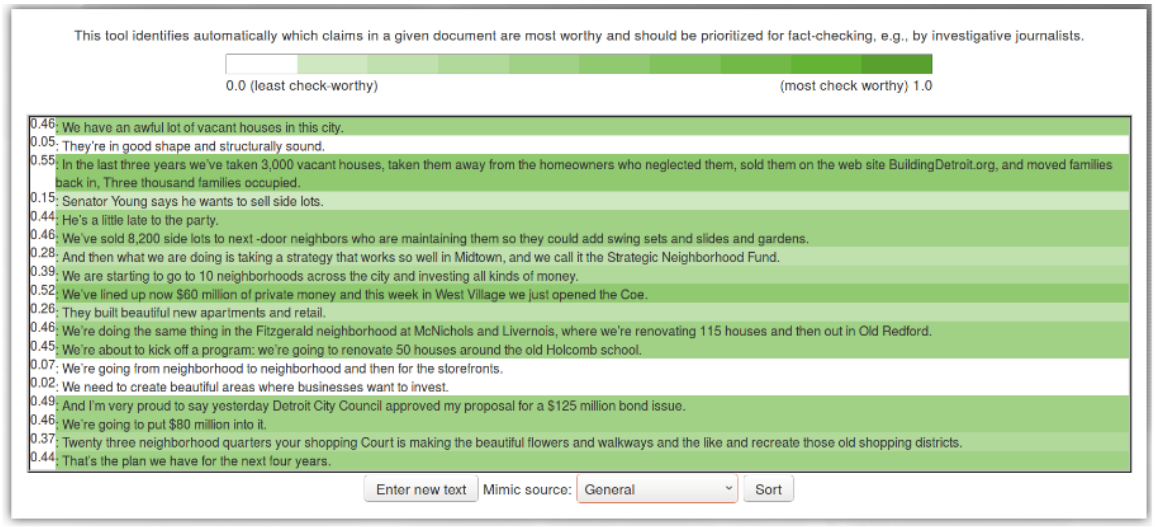


Figure 2-1: An example of the output for an automatic check-worthy claim detection system applied on an *English* presidential debate, in natural order [Jaradat et al., 2018].

.

to the set of evidence that is extracted from reliable sources. A previous work [Baly et al., 2018] used the Google API to retrieve the relevant documents to a claim considered as a query. Then, they re-ranked the documents based on the similarity distance (e.g., cosine similarity between their Term-Frequency (TF) vectors) between the retrieved documents and the claim.

The next step in fact-checking is stance detection as it is important to be able to see whether or not the retrieved documents *agree* or *disagree* with the claim. A recent state-of-the-art work [Mohtarami et al., 2018] used memory networks to perform end-to-end stance detection.

The last step is to decide the factuality of the claim. It is important when checking the factuality of an input claim. In this step, the fact-checking system aims to evaluate the reasoning and use the set of evidence and stances computed from previous steps to predict the input claim as factually true or not. Some works on this front has been conducted on a question answering dataset created on the Qatar Living community forums [Mihaylova et al., 2018].

2.2 Check-Worthy Claim Detection

The focus of this thesis is on the first step: detecting check-worthy claims. Existing datasets for the claim detection task were generated based on published transcript annotations from fact-checking organizations such as Politifact and FactCheck.org [Nakov et al., 2018; Gencheva et al., 2017]. This limits our claims to primarily be a subset of spoken political speeches. Recently, the datasets for other steps of the fact-checking process are created from a more general set of sources, such as the Fact Extraction and VERification (FEVER) dataset [Thorne et al., 2018]. The FEVER dataset consists of claims and the introduction sections of related Wikipedia articles, which are given stance labels of SUPPORTS or REFUTES depending on their relationship with the claim [Thorne et al., 2018]. However, datasets based on a general set of sources have not been annotated for the claim-detection task. In this thesis, we introduce our novel Wikipedia-based dataset that is both much larger and more general than existing datasets for the claim-detection task (Chapter 5).

We aim to build a model upon both previously released datasets and our new Wikipedia dataset that outperforms existing ones such as ClaimRank model. Unlike previous work, our model learns sentence level encodings that takes into account the features that are most relevant for detecting check-worthy claims. These encodings not only provide another avenue for improving existing claim detection models, but also provide additional insight into which features can separate check-worthy claims from non-check-worthy ones in different domains.

In addition, these sentence level encodings should take into account general information about language–the model does not need to be trained on claims labeled with check-worthy labels, instead allowing our model to be pre-trained using datasets much larger than the ones used in previous work. This should allow our model to be generalized to the documents outside politics and to the wider domains, and should allow our model to be learned the language before being optimized for detecting check-worthy claims.

Current models use the average word embeddings to encode an input sentence/claim

[Jaradat et al., 2018]. The ClaimRank model uses pre-trained Word2Vec embeddings as its Word embedding–in general, either pre-trained Word2Vec [Mikolov et al., 2013] or GloVe embeddings [Pennington et al., 2014] are commonly used as word embeddings by previous works. These pre-trained word embeddings include a single embedding for each word and do not reflect the context where the word appears. While, more recent research has focused on create word embeddings that can be updated depending on the context of the word as the meaning of a word is dependent on the context in which it is found. In this thesis, we use these contextual word embeddings to create claim-detection-specific sentence embeddings.

2.3 Contextual Word Embeddings

The two most recent contextual word embeddings that we explore in this thesis are Embeddings from Language Models (ELMo) [Peters et al., 2018] and Bidirectional Encoder Representations from Transformers (BERT) [Devlin et al., 2018]. We will describe the specifics of these models later on in Sections 4.2.1 and 3.2.2 respectively. Both of these models create word embeddings by using neural networks to incorporate every word of an input sentence to determine the output word embedding for each word. For this purpose, ELMo uses biLSTM layers and BERT uses fully connected Transformer layers. Incorporating ELMo and BERT as word embeddings into existing models works extremely well for a variety of natural language classification problems, such as sentiment analysis [Peters et al., 2018; Devlin et al., 2018]. In this thesis, we investigate the effectiveness of these contextual word embeddings for the check-worthy claim-detection task.

Chapter 3

Check-Worthy Claims and Debate Dataset

In this chapter, we focus on our work on the existing Debate dataset [Jaradat et al., 2018]–the dataset is described in Section 3.1. As discussed earlier in Chapter 2, the recent work [Jaradat et al., 2018] on this dataset presented a classifier with a set of hand-crafted features for detecting the check-worthy claims. While we present our neural-based approach with more sophisticated ways to calculate sentence/claim embeddings in Section 3.2. We present our experiments and results on the Debate dataset in Section 3.3, and we finally discuss the effects of transfer learning by looking at how pre-training our sentence embeddings using other datasets–the Politifact dataset (in Chapter 4) and the Wikipedia dataset (in Chapter 5)–can affect the performance on the Debate dataset.

3.1 Dataset Description

The Debate dataset [Jaradat et al., 2018; Gencheva et al., 2017] consists of four different transcribed and annotated debates from the 2016 US presidential election season. Three of the debates are presidential debates, and the fourth is a vice presidential debate. Each of these debates is annotated for factuality by nine different news outlets: ABC News, Chicago Tribune, CNN, FactCheck.org, NPR, PolitiFact, The Guardian,

The New York Times, and the Washington Post. The debates are divided up into sentences, each constitutes one input claim. The check-worthiness label of each claim is determined by combining the labels of the different news outlets. If a single news outlet fact-checked a claim, then the claim is labeled check-worthy; if no news outlet fact-checked a claim, then the claim is labeled not-check-worthy. Out of the 5,415 total sentences in the Debate dataset, 893 of them are annotated and thus considered check-worthy. The statistics of check-worthy sentences for each debate can be seen in Table 3.2, and an example of a labeled section from one of the debates can be seen in Table 3.1.

In Table 3.2, We refer to the first four debates as the DEBATES dataset, which are developed and used in [Jaradat et al., 2018; Gencheva et al., 2017] to make comparison with their results. Also, additional debates and speeches from the 2016 US presidential election season were annotated in the same way and published online by Qatar Computing Research Institute (QCRI). We refer to these additional sources (i.e., the last four rows in Table 3.2) as the SPEECHES dataset. As the table shows, the SPEECHES are much shorter than DEBATES, although SPEECHES and DEBATES are similar sources. In particular, they have similar proportions of check-worthy claims, which makes sense, since debates are essentially platforms for candidates to give mini speeches in succession, whether that be an opening statement or a rebuttal to another candidate's remarks.

3.2 Method

We present models that do not rely on hand crafted features, but instead, use different neural network based approaches to encode the input sentences/claims and generate their embedding vectors. These vectors are then passed to a feed forward neural network to predict check-worthiness. The general architecture of our model is shown in Figure 3-1.

One of the obstacles facing our initial approach is a lack of train/labeled data for this task. This results in an issue that is overfitting to the train data, learning to

Speaker	Number of Sources that Fact-checked	Claim
PENCE	0	For the last seven-and-a-half years, we've seen America's place in the world weakened.
PENCE	3	We've seen an economy stifled by more taxes, more regulation, a war on coal, and a failing health care reform come to be known as Obamacare, and the American people know that we need to make a change.
PENCE	0	And so I want to thank all of you for being – being with us tonight.
PENCE	0	I also want to thank Donald Trump for making that call and inviting us to be a part of this ticket.

Table 3.1: An example from the opening of the 2016 Vice Presidential Debate [Gencheva et al., 2017].

Source	Check-worthy Sentences	Total Sentences	Check-worthy %
First 2016 Pres. Debate	224	1,403	15.97
Vice 2016 Pres. Debate	237	1,358	17.45
Second 2016 Pres. Debate	184	1,303	14.12
Third 2016 Pres. Debate	248	1,351	18.36
Orginal Dataset (DEBATES)	893	5,415	16.49
Ninth 2016 Dem. Pres. Debate	48	1,462	3.28
Trump Rep. Nom. Acceptance Speech	96	374	25.67
Clinton Dem. Nom. Acceptance Speech	44	399	11.03
Trump Pres. Inauguration Speech	17	137	12.41
Additional Dataset (SPEECHES)	205	2,372	8.64
Total	1,098	7,787	14.10

Table 3.2: The statistics of the Debate dataset. We refer to the first four sources as the DEBATES dataset [Gencheva et al., 2017; Jaradat et al., 2018], and to the last four sources as the SPEECHES dataset developed by Qatar Computing Research Institute (QCRI).

memorize the mapping from input to check-worthiness for train data instead of acquiring more general understanding about check-worthy claims. In order to navigate this issue, we can initialize some parts of our model (e.g., encoder part) with training on larger corpora from other domains or similar tasks. This way, our model can begin by knowing various general-level information about words and sentences depending on which weights are initialized based upon an already trained model. This process can be called pre-training the model. Then, the pre-trained model (e.g., encoder and classifier parts) can be fine-tuned on the available small labeled data to encode the domain specific features in order to improve performance.

We investigate the effects of different types of sentence level encoders independently explained in Sections 3.2.1 and 3.2.2. Furthermore, we examine the effectiveness of a combination of our model with a recent previous work [Jaradat et al., 2018]–called ClaimRank model–by concatenating their features with the sentence em-

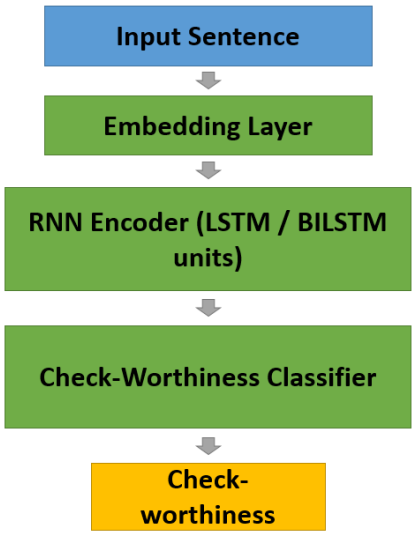


Figure 3-1: The general architecture of our model based on recurrent neural networks. For a given input sentence, our model first takes each word in the sentence and feeds it to a pre-trained 300-dimensional GloVe embedding [Pennington et al., 2014] and concatenates the embeddings. The concatenated embedding vector becomes the input to an LSTM [Hochreiter and Schmidhuber, 1997] or biLSTM [Graves and Schmidhuber, 2005] layer. The final output of the LSTM or biLSTM layer becomes the input to a feed-forward neural network classifier, whose output is the predicted check-worthiness of the input.

bedding vector from our model and then passing the concatenation vector to a feed forward layer to detect the check-worthy claims (see Figure 3-2).

3.2.1 LSTM and biLSTM

To encode the input sentences/claims, we use Recurrent Neural Networks (RNNs)– e.g., Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997] and bidirectional LSTM (biLSTM) [Graves and Schmidhuber, 2005]. In general, RNNs model sequences where there is information to be gained by the interaction between inputs. In our case, the meaning of a sentence depends on the interaction of their words, so we use RNNs to help model this interaction and create a vector for the entire sentence. However, in a given sentence, some pieces of it, such as the subject and the verb, are more important than others and need to be maintained for the entirety of the sentence. LSTMs are better at this task than RNNs because LSTMs

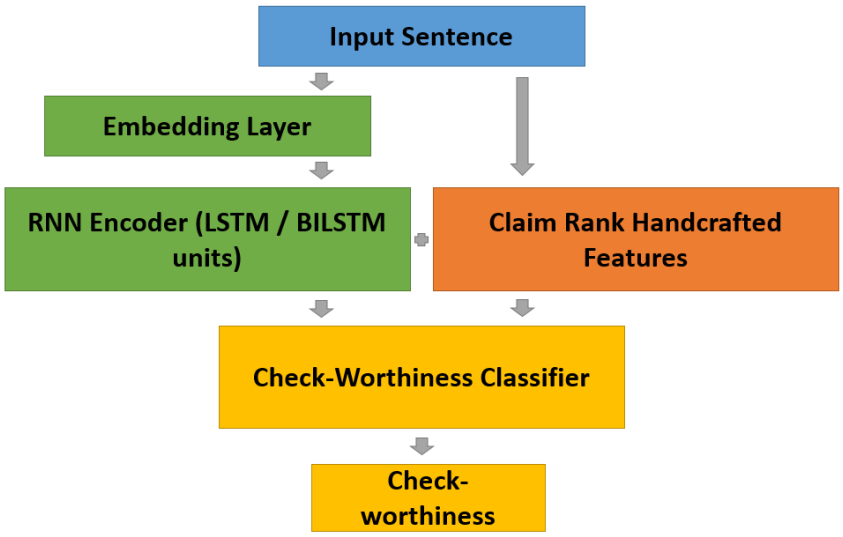


Figure 3-2: The general architecture of a combination of our model and ClaimRank Model [Jaradat et al., 2018]. For an input sentence, our model first takes each word in the sentence and feeds it to a pre-trained 300-dimensional GloVe embedding [Pennington et al., 2014] and concatenates the embeddings. The concatenated embedding vector becomes the input to an LSTM [Hochreiter and Schmidhuber, 1997] or biLSTM [Graves and Schmidhuber, 2005] layer. In parallel, our model computes the set of handcrafted features from the ClaimRank model for our input sentence and concatenates them into a feature vector. A final concatenated vector of the final output of the LSTM or biLSTM layer and the ClaimRank feature vector becomes the input to a feed-forward neural network classifier, whose output is the predicted check-worthiness of the input.

have built-in memory units to maintain information for long inputs [Hochreiter and Schmidhuber, 1997]. The biLSTMs built upon this concept by combining a forward LSTM with a backward LSTM to track the interaction between inputs and those that both precede and follow it [Graves and Schmidhuber, 2005].

We start our approaches with these RNN units because they are two of the simplest recurrent models and have the least amount of parameters to train. In addition to being extra baselines to compare against, we aim to incorporate our claim-detection models into a fact-checking pipeline, so speed of our presented model is a consideration.

We use the 300-dimensional GloVe word embeddings trained on Wikipedia and the Gigaword 5 database [Pennington et al., 2014] to first map words to their pre-trained embeddings and then use the RNN units to generate embedding vectors for input sentences. These word embeddings can be updated during training of our model. The advantage of having these embeddings be trainable is that they can be shifted to incorporate our particular claim-detection task; however, this potential gain comes at the cost of heavily increasing the number of trainable parameters and the training time.

3.2.2 BERT

We further use the Bidirectional Encoder Representations from Transformers (BERT) [Vaswani et al., 2017] to encode the input sentences/claims. It aims to create the representations that takes into context the entirety of a sentence when creating word embeddings. For example, a word that its embedding should be changed depending on context can be the word *bank*. It could be either a noun or verb depending on the context as in the following two sentences:

- Do not *bank* on the bus arriving early when planning ahead.
- I went to the *bank* in order to withdraw money.

The Transformers in BERT learn context entirely through a self-attention mechanism instead of through recurrence, which is how LSTMs and biLSTMs do it [Vaswani

et al., 2017]. A model of a transformer is presented in Figure 3-3. In the figure, the transformer has two parts; an encoder that only uses a self-attention mechanism (left) and a decoder, which uses a similar self-attention mechanism except that it also takes as input the output of the decoders from the previous layers.

BERT embeddings as standalone models have been shown to increase performance on a variety of Natural Language Processing (NLP) tasks [Devlin et al., 2018]. The reason is that BERT's transformer layers are fully connected, every node and in particular the output nodes are calculated with the entirety of the last hidden layer as context. Thus, each output node is fully dependent on all of the inputs. Thus, the BERT model can be directly utilized for classification tasks and the output corresponding to the first input token–which is specified as a special [CLS] symbol–can be used as a sentence level embedding vector [Devlin et al., 2018]. A graphical representation of using this special [CLS] token for classification is presented by Devlin et al. [2018] and shown in Figure 3-4.

In our experiments, we initialize our model with the uncased $\text{BERT}_{\text{base}}$ model[1]. We then fine-tune our model using only the original four debates in the DEBATES dataset that used to train the baseline ClaimRank model. We experiment with different settings in order to examine how the BERT model would fine-tune itself on the DEBATES dataset.

3.2.3 Model Settings

We use the Adam optimizer to minimize the binary cross-entropy loss, as we only have two classes. The hyper-parameters, especially the learning rate and batch size, are chosen to increase the stability of our model and avoid overfitting of the model. In order to do this, we decrease both the learning rate and batch size until the model's performance on the validation set improves during training, at least until epoch five or so.

For each model configuration, we perform a 4-fold cross-validation. In each fold,

[1] `https://github.com/huggingface/pytorch-pretrained-BERT` is where we got our pre-trained weights.

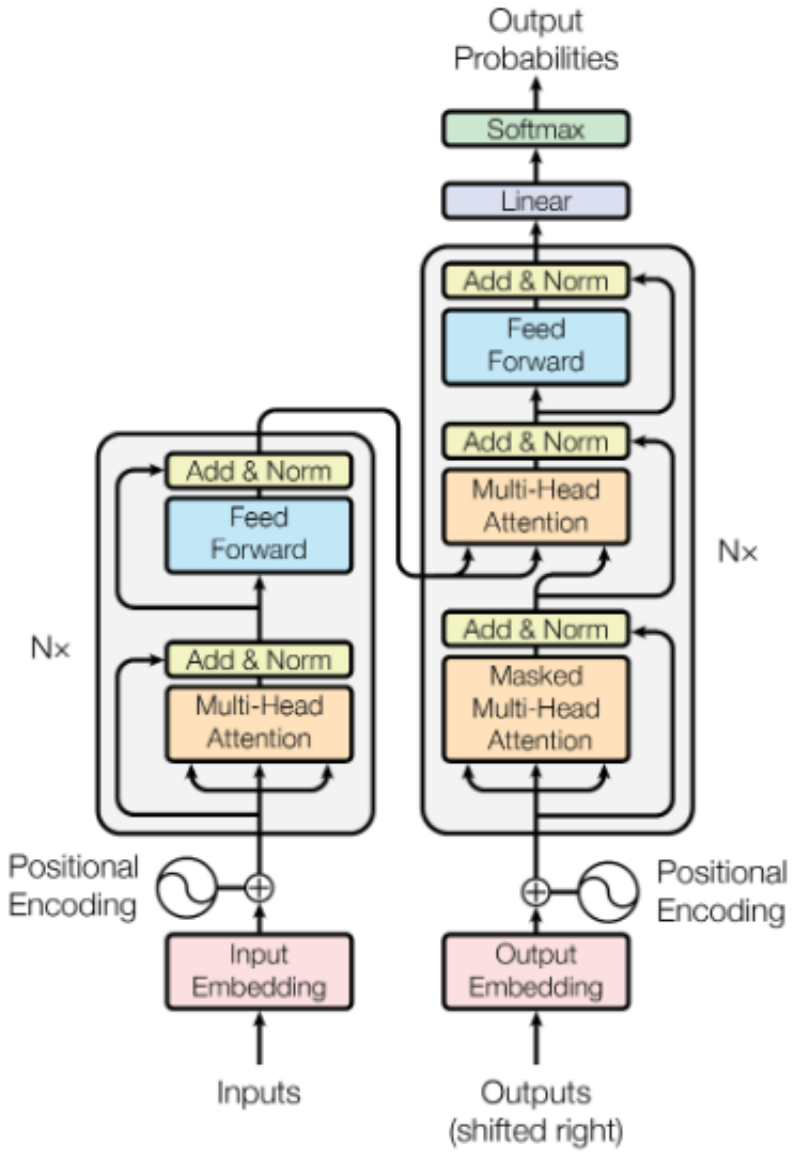


Figure 3-3: A model of a transformer as presented by Vaswani et al. [2017]. The transformer has two parts, an encoder that only uses a self-attention mechanism (left) and a decoder, which uses a similar self-attention mechanism except that it also takes as input the output of the decoders from the previous layers.

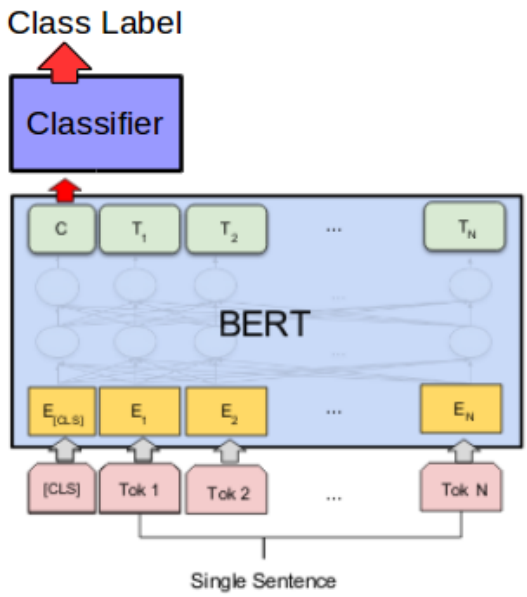


Figure 3-4: A model of using BERT as part of a classification problem presented by Devlin et al. [2018]. In this scheme, the input sentence is tokenized with a special [CLS] token placed at the head of the sentence. Then, the tokens are simultaneously passed into the BERT model, and finally the first output node corresponded to the [CLS] token is used as input to the classifier.

we hold out one debate as the test set and one debate as the validation set. The remaining two debates are used as the train set. When we incorporate the SPEECHES dataset, we use the same experimental set-up except that we add the entirety of the SPEECHES dataset to the train set.

3.3 Results and Analysis

3.3.1 Evaluation Metrics

Due to the severely unbalanced nature of our dataset, we focusd on evaluation metrics such as macro-F1, equally weighted across the check-worthy and not-check-worthy classes, and average precision. Average precision for a single debate is calculated as follows:

$$\text{average precision} = \frac{\sum_{k=1}^{n}(P(k) \times rel(k))}{\text{number of check-worthy claims}} \tag{3.1}$$

where $P(k)$ is the precision at k and $rel(k)$ is 1 if the kth claim, as determined by ranking the claims by the check-worthiness score given to them by the model, is check-worthy and 0 otherwise. Average precision is a useful metric to use because of limitations regarding the completeness of the Debate dataset. We know for certain that all the claims annotated as check-worthy claims are the most check-worthy of claims. However, it is highly likely that some of the claims labeled not-check-worthy are actually check-worthy, but are not fact-checked due to logistical issues such as journalistic manpower or impending deadlines. For example, the following two claims have not been fact-checked by any of the journalist organizations:

- "My grandfather had immigrated to this country when he was about my son's age." -Pence (2016 Vice Presidential Debate)
- "He left his state about $2 billion in the hole." - Pence (2016 Vice Presidential Debate)

The first claim is potentially check-worthy because immigration was such a big topic during the 2016 election cycle, and the second claim is potentially check-worthy because Pence is making a claim as to Kaine's incompetence as an economic leader.

For these reasons, average precision is the metric that we use to compare model performance. However, we still would like to consider the class F1 scores and macro-F1, average of the two class F1 scores, because our dataset is so unbalanced. This allows us to get a closer look into how the unbalance of our dataset is affecting our models performances.

3.3.2 LSTM and biLSTM

Our results can be seen in Table 3.3. These results are the average performance of our models on the test sets across 4-fold cross-validation on the Debate dataset. When (S) is present in the table, the train set also included transcripts from the SPEECHES dataset.

In the table, the first row consists of the performance of the previous work–ClaimRank model–reported in the original paper [Gencheva et al., 2017]. The second

Model	AvgP	PR@5	PR@10	PR@20	PR@50	R Precision
1. ClaimRank Paper	42.70	80.00	72.50	71.25	60.00	43.20
2. ClaimRank Paper Replicated	41.80	80.00	77.50	65.00	56.00	42.47
3. Dyanmic LSTM Encoding	28.85	35.00	47.50	41.25	36.00	32.22
4. Dynamic biLSTM Encoding	30.34	65.00	50.00	45.00	41.00	31.78
5. Fixed LSTM Encoding	31.31	55.00	57.50	43.75	46.00	33.28
6. Fixed biLSTM Encoding	30.34	65.00	55.00	47.50	41.50	32.57
7. Fixed LSTM + ClaimRank	38.11	75.00	72.50	62.50	53.00	38.33
8. Fixed biLSTM + ClaimRank	36.94	65.00	60.00	56.24	48.00	39.34
9. Dyanmic LSTM Encoding (S)	35.56	65.00	52.50	52.50	50.50	36.96
10. Dynamic biLSTM Encoding (S)	35.52	60.00	52.50	51.25	48.00	37.30
11. Fixed LSTM Encoding (S)	35.46	60.00	57.50	55.00	53.50	36.06
12. Fixed biLSTM Encoding (S)	34.33	70.00	55.00	47.50	42.00	35.16
13. Fixed LSTM + ClaimRank (S)	42.66	80.00	80.00	71.25	57.50	41.62
14. Fixed biLSTM + ClaimRank (S)	42.23	75.00	72.50	65.00	55.50	43.00

Table 3.3: Results of models trained on the Debate dataset. The results are the average of each metric across all four cross-validation folds. When (S) is present, the training set also included transcripts from the SPEECHES dataset. "R Precision" is defined as the precision at R and R equals to the number of check-worthy claims in the source.

row shows our result when we replicate the experiments of the ClaimRank model. As we can see, our reported performance is very similar to the paper's reported performance with the exception of a slightly lower average precision.

Rows 3–8 show the results of our experiments when only training on the original DEBATE dataset. Rows 9–14 show the results of our experiments when we include the additional four transcripts of the SPEECHES dataset. We will discuss these results in more detail below.

LSTM vs biLSTM

Rows 3–4 and 9–10 show the results of our experiments when comparing dynamic LSTM models to dynamic biLSTM models, and rows 5–6 and 11–12 show the results of our experiments when comparing fixed LSTM to fixed biLSTM models. It is clear from these results that using biLSTM instead of LSTM does not affect performance much. For both pairs of experiments, the average precision is within 1.5% when changing from LSTM to biLSTM. These results indicate that there is not much to be gained by looking at a sentence bidirectionally as opposed to unidirectionally.

Fixed vs dynamic word embeddings

Comparing rows (3, 5), (4, 6), (9, 11) and (10, 12), we can see that there is minimal difference between shifting from a dynamic encoding to a fixed encoding. These results show that the Debate dataset has a negligible impact on affecting pre-trained word embeddings. This could be because the dataset is so small in comparison to what the original embedding was pre-trained on, so the extra information gained is essentially irrelevant.

In addition, the embeddings that we are attempting to train are word embeddings. In the context of political debates and speeches, it is important to be easily understood by a large audience, so the language used is not going to be highly idiosyncratic. This can explain why our performance does not change much when going from dynamic to fixed encodings as pre-training on our dataset is probably not going to shift the word embeddings much.

Addition of ClaimRank features

When we compare rows (5, 7), (6, 8), (11, 13) and (12, 14), we see that the average precision increases by about 7% when combining the ClaimRank features to our encoder features. This shows that the ClaimRank features contains relevant information that is not currently being encoded by our model. When we compare rows (8,9) with row (2), we see that this combination actually brings the performance down when compared to the original ClaimRank implementation.

The original ClaimRank implementation incorporates a basic sentence level embedding using the average of 300-dimensional word2vec [Mikolov et al., 2013] embeddings pre-trained on Google News for the words in an input sentence. Although we expect that an RNN based embedding should encode more information than average word vectors, the drop in performance can be attributed to the effectiveness of using Word2Vec word embeddings over the GloVe word embeddings used in the RNN.

Addition of SPEECHES dataset

When comparing rows 3–8 to rows 9–14, we can see that adding in the additional SPEECHES dataset used for training helps to improve performance across the board by about 4% to 5% on mean average precision. As we have inferred earlier in our comparison of fixed embeddings to dynamic embeddings, these improvements are probably not due to fine-tuning our word embeddings to our dataset. Even with the addition of the SPEECHES dataset, the total size of our dataset is miniscule in comparison to the size of the dataset that our GloVe and Word2Vec embeddings are trained on. This implies that primary benefit of adding in the SPEECHES dataset comes from being able to better fine-tune the final classifier. Interestingly enough, adding in more data improves our overall performance even though adding the SPEECHES dataset makes the imbalance issue even worse.

3.3.3 BERT

Our experimental results for BERT with different settings are shown in Table 3.4. We can immediately see by comparing rows 2–14 to row 1 that BERT causes our model to be better at detecting not-check-worthy claims than ClaimRank, but be worse at detecting check-worthy claims. The reason is most likely due to importance of context in determining the check-worthiness of a claim. The ClaimRank model considers the features that attempt to calculate the relationship between a claim and its surrounding context. While our BERT model only takes a single sentence as input, so even though the model is better equipped to learn the nuisances of the claim than ClaimRank, it is unequipped to handle claims whose check-worthiness is dependent on the claims around it.

Overall, rows 5–6 and 11–13 have average precisions that are comparable or better than row 1, our replication of the ClaimRank model. BERT is the best standalone sentence level encoder as it has a better average precision than all the standalone sentence level encoders in Table 3.3. Our best model, row 11, has a mean average precision 1.7% better than the ClaimRank model despite not taking into account any

Model	AvgP	F1	F1 Neg.	F1 Pos.	R Precision	Accuracy
1. ClaimRank Paper Replicated	41.81	63.50	83.40	43.61	42.47	74.53
2. BERT [3 epochs]	39.58	61.45	89.72	33.18	42.50	82.32
3. BERT [5 epochs]	40.27	63.38	89.74	37.02	41.50	82.36
4. BERT [10 epochs]	38.82	63.85	89.94	37.76	40.99	82.71
5. BERT [20 epochs]	41.75	64.51	90.06	38.96	43.21	82.93
6. BERT [100 epochs]	41.43	64.53	90.26	38.80	43.97	83.20
7. BERT [200 epochs]	40.46	64.56	90.34	38.78	43.32	83.33
8. BERT [10 epochs][Adam]	32.59	58.73	88.71	28.74	36.39	80.64
9. BERT [20 epochs][Adam]	32.65	57.54	88.89	26.20	34.49	80.72
10. BERT [100 epochs][Adam]	29.77	58.13	88.13	28.12	31.27	79.66
11. BERT [20 epochs][lr 1e-5]	43.49	64.91	90.53	39.29	45.57	83.65
12. BERT [20 epochs][warm 0.5]	41.88	64.05	90.31	37.78	44.93	83.26
13. BERT [20 epochs][lr 1e-5][warm 0.5]	42.10	64.47	90.33	38.61	44.73	83.33
14. BERT [10 epochs][Adam][lr 1e-5]	41.89	63.29	90.03	36.55	43.30	82.87
15. BERT [pre-trained][100 epochs]	34.61	60.24	89.84	30.64	39.47	82.30

Table 3.4: The experimental results for BERT with settings. Our positive class is the check-worthy class, and our negative class is the not-check-worthy class. The default optimizer is the BERT-Adam optimizer, the default learning rate is 5e-5, and the default warming-up proportion is 10% of the training epochs. Row 15 is first pre-trained on the Wikipedia dataset.

contextual features.

Comparing Learning Rates

When we compare rows 8–10 with rows 4—6, we can immediately see that the BertAdam optimizer with the default settings of warm-up for the first 10% of the training epochs is the superior choice to the standard Adam optimizer. The biggest difference between the two optimizers is that our BertAdam optimizer's learning rate starts by following a linear warm-up schedule. As we can see in Figure 3-5 and by comparing rows 3–7 in Table 3.4, all our models are prone to overfitting as our dataset is so small in comparison to the number of parameters in the model. Empirically, comparing row 8 with 14 shows that our model is very sensitive to perturbations. 5e-5 is a fairly small learning rate, yet our parameters still move too much in our gradient descent step, so the resulting fitted model has a higher loss and lower average precision, by around 9%, than the model fitted with the smaller 1e-5 learning rate. This is why our models with either learning rate warm-up or smaller learning

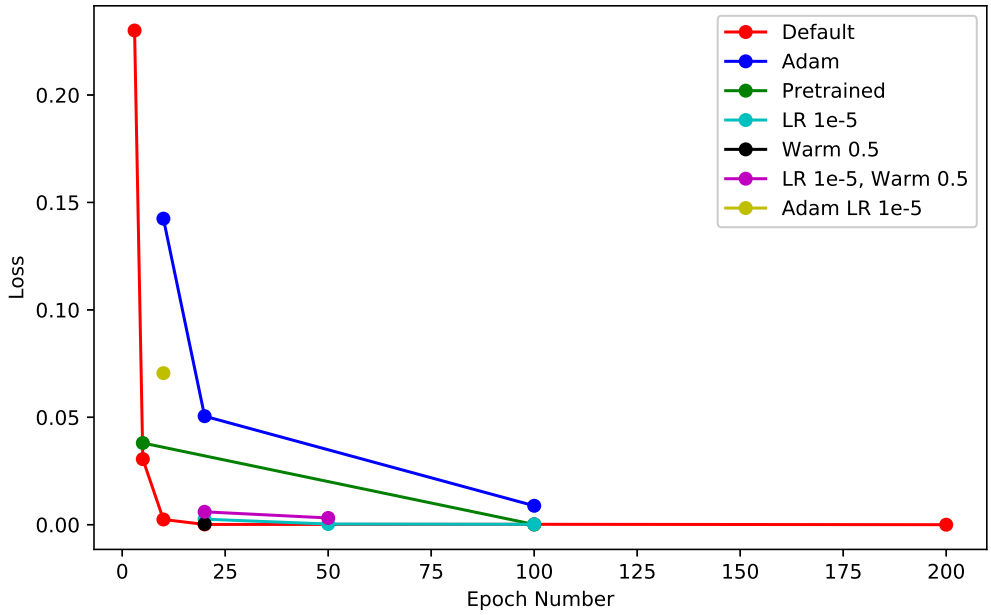


Figure 3-5: BERT training losses for different settings through different epochs. Unless otherwise stated, the default learning rate is 5e-5, the default warm-up proportion is 0.1, and the default optimizer is the BERT optimizer. The pretrained model was pretrained on the Wikipedia dataset, while the other models were only trained on the Debates dataset.

rates–rows 1–7 and 11–14–perform so much better than rows 8–10.

Pre-training on Wikipedia

When comparing rows 14 with 6, we can see that the model that is first pre-trained on the Wikiedia dataset–explained in Chapter 5–performs far worse than the one that is only fine-tuned on the Debate dataset. Our dataset is so small and a variety of small reasons can lead to a noticeable drop in performance. One possible reason is that training our uncased $\text{BERT}_{\text{base}}$ model on our Wikipedia dataset can lead our model picking up nuances about claims in our Wikipedia claims dataset that are not present in our Debate dataset as the two have very different contexts. In particular, our Wikipedia dataset covers the entire spectrum of topics whereas our Debate dataset only covers the 2016 presidential election cycle. In addition, our Wikipedia dataset consists of written information whereas our Debate dataset consists of spoken communication, which have different idiosyncrasies. These factors in conjunction may explain why the off the shelf uncased $\text{BERT}_{\text{base}}$ model trained by Google is a better starting point than pre-training on our Wikipedia dataset.

3.4 Summary

In summary, we have shown that for the claim-detection task, there is a lot of information encoded in the context of a sentence that is relevant for detecting check-worthy claims. Claim-detection can be done for an input sentence, while it is necessary to consider the context (document) in which the sentence is found in order to get maximum performance. Based on our experiments with RNNs and BERT, we have also shown that a BERT model fine-tuned with a very low learning rate on our dataset is the most effective sentence level encoder. Overall, we have shown that carefully choosing the sentence encoder can lead to improvements in the claim-detection task, but the improvements are minimal at best as the Debate dataset is imbalanced, and its small size makes overfitting a constant issue that needs to be overcome.

Chapter 4

Check-Worthy Claims and Politifact Dataset

One of the major shortcomings with the Debate dataset, as shown in Table 3.2, is its imbalance between the check-worthy and not-check-worthy classes. In particular, there are 67% more not-check-worthy claims than check-worthy claims in the original DEBATES dataset. In this chapter, we attempts to incorporate a dataset taken from the fact-checking website Politifact in order to balance the classes, described in Section 4.1. We then present our model and experiments to show the effectiveness of the additional new dataset in Sections 4.2 and 4.2.2. Finally we summarize our findings about the differences between the models trained with Politifact (Chapter 4) and models trained only with the Debate dataset (Chapter 3).

4.1 Dataset Description

A description of the Politifact dataset is shown in Table 4.1. The dataset consists of 13,084 claims, most of them are in the form of a quotation, alongside various meta information about the claim. Politifact prioritizes using primary sources and original documentation as the evidence for a claim's Truth-o-Meter score[1]. For each

[1] https://www.politifact.com/truth-o-meter/article/2018/feb/12/principles-truth-o-meter-politifacts-methodology-i/

quotation, we attempt to find transcripts that contain the quote and the context where it actually appeared in. These sources are always listed at the side of published article–an example can be seen in Figure 4-1[2].

As Table 4.2 shows, all the claims in our Politifact dataset are by definition check-worthy because they are chosen by Politifact to be fact-checked. This allows us to incorporate a large number of positive check-worthy labeled claims to the original Debate dataset to help balance the two classes. As Table 4.2 shows, incorporating the Politifact dataset has the effect of balancing the two class labels.

As the table alludes, we do not use the entirety of the Politifact dataset. The primary reason for this is that we are not always able to find the context from which a claim originated. As we have seen in Chapter 3, the context surrounding a claim is important for determining check-worthiness. Therefore, we decided to use only the claims for which we are able to find the original source among the list of sources provided by Politifact. One issue with this is that the language, structure and style of the Debate and Politifact datasets may be different since the Debate datasets consist entirely of transcripts of spoken communication, whereas the Politifact dataset consists of quotes that are written like blog posts or news articles. Thus, the context for a claim in the Debate dataset is always other quotes, whereas the context for a claim in our Politifact dataset is often written commentary or news reporting.

4.1.1 Politifact (Exact)

We consider a subset of claims from the Politifact dataset that includes the claims for which we are able to find the exact claim in one of the sources. An example is shown in 4-2[3] [4]. For these claims, we use the source article when considering the context for all feature extraction purposes. Table 4.2 shows this subset is smaller than the two datasets as most claims are paraphrases of claims found within the source articles

[2] `https://www.politifact.com/health-check/statements/2019/may/13/eric-swalwell/eric-swalwells-tweet-about-georgias-new-abortion-r/` Source for the image.

[3] `https://www.politifact.com/ohio/statements/2016/aug/12/ohio-right-life/ohio-right-lifes-faulty-claims-ted-strickland-abor` contains the claim.

[4] https://www.lifenews.com/2016/08/02/ohio-ted-strickland-wants-to-force-ohio-taxpayers-to-fund-abortion contains the source.

"The so-called 'heartbeat' law outlaws abortion before most women even know that they're pregnant. This is one of the most restrictive anti-abortion laws in our country."

— *Eric Swalwell on Wednesday, May 8th, 2019 in posted on Twitter*

MOSTLY TRUE

Swalwell's tweet about Georgia's new abortion law is only slightly off-key

By Shefali Luthra on Monday, May 13th, 2019 at 9:09 a.m.

Georgia Gov. Brian Kemp recently signed into law the state's latest abortion restriction. Political reaction to the measure, which prohibits the procedure once a doctor can detect a fetal heartbeat — usually at about the six-week mark — was swift.

About this statement:

Published: Monday, May 13th, 2019 at 9:09 a.m.

Researched by: Shefali Luthra

Edited by: Stephanie Stapleton

Subjects: Abortion, Health Care

Sources:

Rep. Eric Swalwell (D-Calif.), Twitter post, May 8, 2019.

The Atlanta Journal-Constitution, "Kemp Signs Anti-Abortion 'Heartbeat' Legislation, Sets Up Legal Fight," May 7, 2019.

Guttmacher Institute, "A Surge in Bans on Abortion as Early as Six Weeks, Before Most People Know They Are Pregnant," updated May 7, 2019.

Figure 4-1: An example Politfact page. The claim is in rectangle at the top and is in the form of a quotation. The sources supporting the Truth-o-Meter label of Mostly-True as well as the original source of the claim can be found on the right of the page. In this case, the source of the claim is a Tweet, which is listed without a link at the top of the sources area. Though the other sources are about the new law, neither directly quote the claim.

rather than exact matches.

4.1.2 Poltifiact (Similar)

In addition, we consider a subset of claims from the Politifact dataset that includes the claims for which we are able to find a sentence that is lexically similar to the claim in a source article. An example is shown in 4-3[5] [6]. Similar to the *Exact* dataset, we use the source with the similar claim to extract context based features. In addition, we use the sentence found in the source article similar to an input claim as the claim itself rather than the quotation listed on the Politifact website. This dataset is a strict superset of our exact dataset as exact matches are also lexically similar matches. Thus, during our experiments, we do not use both datasets at the same time in order to avoid duplicating claims.

[5] `https://www.politifact.com/ohio/statements/2013/jul/09/phil-burress/phil-burress-says-all-major-religions-are-against-` contains the claim.

[6] `https://www.cleveland.com/open/2013/06/ohio_sorts_out_impact_of_gay_m.html#/0` contains the source.

"Ted Strickland stands in lockstep with Hillary Clinton and the abortion lobby in their desire to force Americans to pay for abortion on demand, up until the moment of birth, with their taxpayer dollars."

— *Ohio Right to Life on Tuesday, August 2nd, 2016 in a press release*

MOSTLY FALSE

Ohio Right to Life president Mike Gonidakis and SBA List president Marjorie Dannenfelser released the following comment:

"Ted Strickland stands in lockstep with Hillary Clinton and the abortion lobby in their desire to force Americans to pay for abortion on demand, up until the moment of birth, with their taxpayer dollars. He's made clear today he will fight for Big Abortion, not Ohio families – a position that is profoundly unpopular – including with members of his own party. Already, four Senate Democrats have stated their opposition to the repealing of Hyde including three Senators – Joe Manchin (WV), Bob Casey (PA), and Joe Donnelly (IN) – who represent neighboring states to Ohio.

Figure 4-2: An example of a claim in our *Exact* dataset. The top is the claim as found on Politifact, and the bottom is the source where the exact match is being highlighted.

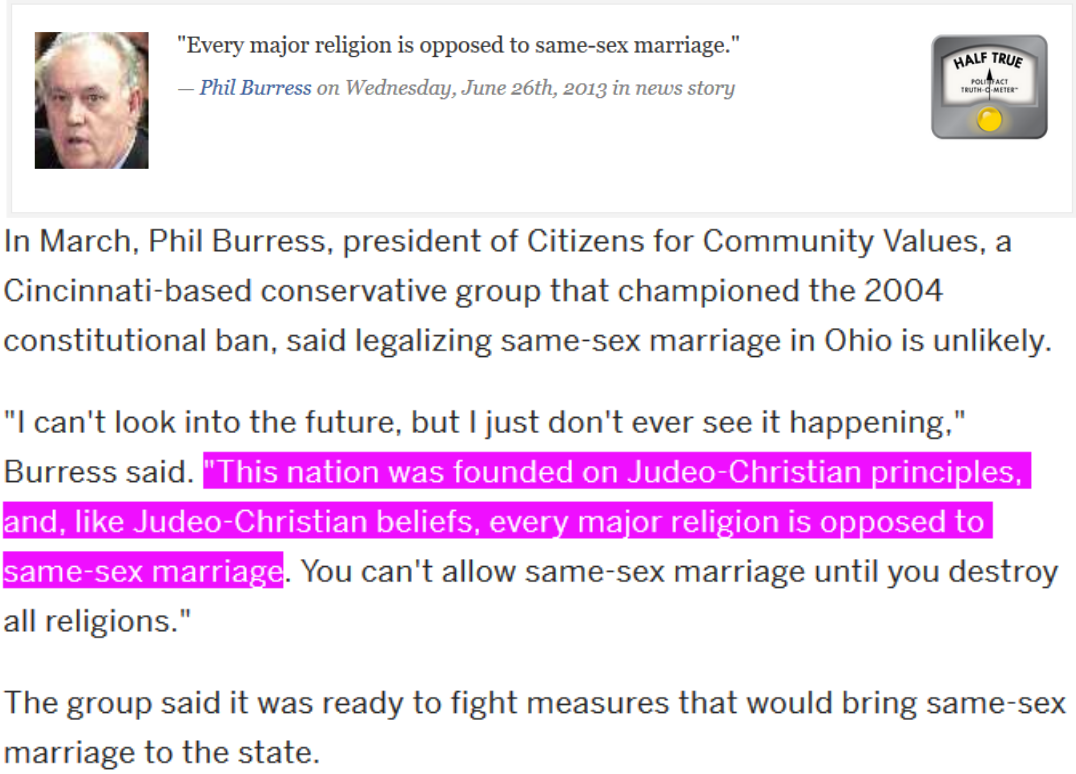


Figure 4-3: An example of a claim in our *Similar* dataset. The top is the claim as found on Politifact, and the bottom is the source where lexically similar match is being highlighted.

Source	Check-worthy Sentences	Total Sentences	Check-worthy %
Politifact	13,084	13,084	100.00
Politifact (Exact)	538	538	100.00
Politifact (Similar)	2,146	2,146	100.00
DEBATES	893	5,415	16.49
DEBATES + Exact	1,431	5,953	24.04
DEBATES + Similar	3,039	7,561	40.19
SPEECHES	205	2,372	8.64
SPEECHES + Exact	743	2,910	25.53
SPEECHES + Similar	2,351	4,518	52.04
DEBATES + SPEECHES + Exact	1,636	8,325	19.65
DEBATES + SPEECHES + Similar	3,244	9,933	32.66

Table 4.1: Statistics of our Politifact dataset. The claims in the *Exact* portion of the dataset are ones where at least one of the sources contains an exact match for the claim. The claims in the *Similar* portion of the dataset are ones where at least one of the sources contains a sentence that is lexically similar to the claim.

4.2 Method

In this chapter, we use the same feed-forward neural network architecture as in Chapter 3. We perform our experiments with a different training set and different ways to create feature vectors. As we mention above, our Politifact dataset, though created based on the journalist labels as our Debate dataset, has an issue regarding source type (i.e., written commentary or news reporting) that could make it different enough from the Debate dataset. Thus, as a first approach, we use a transfer learning approach by first pre-training on Politifact dataset and then fine-tuning on the Debate dataset.

Our second approach ignores the differences and just combines the Politifact dataset into our train set as additional positive examples. We evaluate its effectiveness via the basic ClaimRank model and via a contextual word embedding ELMo model described at following section.

4.2.1 ELMo

ELMo is a word embedding model that attempts to take into account the context in which a word is found. A graphic of the ELMo model structure is shown in Figure 4-

4[7]. The ELMo model uses two biLSTM layers with nodes equal to the number of words in the input sentence to capture the sentence context for a word [Peters et al., 2018].

Then, an output from a node of the biLSTM corresponds to a word embedding, and the embeddings from the layers are concatenated together to create the final embedding vector for the word. This is because each of the layers has learned something different about the word[8]. Our ELMo word embedding is initialized with weights pre-trained on a Billion Word Benchmark[9].

Finally, the vector representation of a sentence/claim is the concatenation of its individual word vectors. This sentence representation is used as the input to our final classifier.

4.2.2 Experimental Settings

In our experiments with the ClaimRank feature set, we vary the train set to compare the effects of our Politifact dataset. All our experiments include the DEBATES dataset. We look at four different variations to add to our train set: nothing,

[7] `https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html` is the source of the graphic.

[8] https://www.slideshare.net/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018

[9] The weights can be found at `https://tfhub.dev/google/elmo/2`.

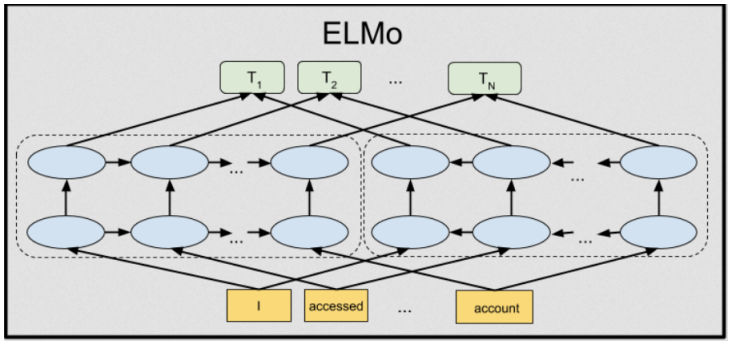


Figure 4-4: A representation of the ELMo structure created by Devlin et al. [2018]. Each word in the input sentence is fed through a bidirectional LSTM that takes the entire sentence as an input in order to create the output word embedding.

SPEECHES, Similar, and Exact[10].

For our transfer learning experiments, we need to augment our pre-train dataset with the SPEECHES dataset because Politifact dataset by itself consists of only positive examples, and it is not straightforward to create negative examples from the sources.

For our ELMo model, we perform experiments with different datasets to fine-tune our model, which has been initialized with weights learned from a Billion Word Benchmark dataset. We incorporate the DEBATES and SPEECHES datasets in all our experiments. We look at the effects of the following three datasets: nothing, Similar, and Exact.

4.3 Results and Analysis

The results of our various experiments are shown in Table 4-1. When comparing row 1 with rows 2–10, none of our models are able to outperform the original ClaimRank implementation trained on the basic DEBATES dataset. The main reason is the language and context of the Politifact dataset are different from Debate dataset. The results are explained in details below.

4.3.1 Adding Similar vs Exact to training set

Comparing rows 1–4, we can see that using Similar and Exact datasets drastically reduce performance, dropping mean average precision by more than 10% when compared to using nothing or the SPEECHES dataset. Table 4 shows that the SPEECHES dataset proportionally has 8% less check-worthy claims than the DEBATES dataset, so adding the SPEECHES dataset changes the train set by decreasing the proportion of check-worthy claims. As our experiments are evaluated based on the DEBATES dataset, the decrease in performance from row 1 to row 2 could be due to the differences in label proportion between the two experiments. This shows that our classifier

[10]SPEECHES is from our Debate dataset, and Similar and Exact are our two Politifact subsets.

is potentially biased towards the majority class in the train set, while it is training to identify check-worthy claims.

The drastic drop in performance between row 1 and rows 3 and 4 is due in part to the structure/language of the claims in the Politifact dataset. As mentioned in Section 4.1, the Politifact claims are taken from articles instead of from speech or debate transcripts. As we showed in Section 3.3, contextual features are important parts of the ClaimRank model, but the context for Politifact claims are very different than the context for claims in the Debate dataset. In particular, for Debate claims, the context is given by the same speaker as the claim, whereas the context for Politifact claims, as in Figure 4-2, may not be. These differences may confound the way our classifiers weigh the contextual features.

4.3.2 Pre-training on Similar vs Exact

Differences in the Similar and Exact datasets vs the DEBATES and SPEECHES datasets being the cause of our drops in classifier performance is shown by rows 5 and 6. Pre-training on these Politifact datasets also causes similar drops in classifier performance. The reason is that initializing our classifier for classifying check-worthy claims in our Politifact datasets causes our classifier to get stuck in a local minimum that is worse than just random initialization. This implies that the features which work best for our Similar and Exact datasets are very different than the ones that work for our Politifact dataset, which is consistent with there being major differences between the different datasets.

4.3.3 Training ELMo on Similar vs Exact

Rows 8–10 further show that the main difference between the two datasets comes from the differences in context as ELMo creates embeddings only using the current sentence. Thus, the ELMo embeddings depend only on the claims themselves–which are similar between the different datasets as all the claims contain quotations–and not the surrounding claims.

Comparing rows 8–10 to row 1 reiterates the importance of context as all the ELMo based models have mean average precisions that are at least 4% less than the ClaimRank model. When we compare Table 4.2 to Table 3.4, we notice that all the word embedding models perform worse on the positive examples than the ClaimRank model. This gives evidence that some claims are check-worthy because of their surrounding context which neither of the embedding models are able to pick up.

Row 7 shows that including the ELMo vectors improves the performance of the model on the not-check-worthy class, but not on the check-worthy class. Row 7 has a lower F1 on the positive class than row 1 but a higher F1 on the negative class. This implies that context is not as important for identifying not-check-worthy claims as the word embedding vectors add in additional information about the claim itself.

We see the same drop in F1 on the positive class but rise in F1 on the negative class when we compare rows 9 and 10 to row 1. This further emphasizes how the information added by the ELMo embedding vectors is more useful for identifying not-check-worthy claims than for identifying check-worthy claims.

4.4 Summary

In summary, we have shown that the differences in context between written claims–as in our Politifact dataset–and spoken claims–as in our ClaimRank dataset– that negatively impact generalizing from a dataset of written claims to a dataset of spoken claims. We have also shown that context is more important for identifying check-worthy claims than not-check-worthy claims as the addition of better word embeddings as features without using much context improves detection of not-check-worthy claims, while decreasing detection of check-worthy claims.

Model	AvgP	F1	F1 Neg.	F1 Pos.	R Precision	Accuracy
1. ClaimRank	41.81	63.50	83.40	43.61	42.47	74.53
2. ClaimRank (SPEECHES)	41.73	58.73	77.27	40.19	42.21	68.16
3. ClaimRank (Similar)	28.36	49.00	63.40	34.60	30.48	53.67
4. ClaimRank (Exact)	34.78	50.34	63.08	37.60	36.27	53.7
5. ClaimRank [SPEECHES, Similar]	27.51	52.87	90.33	15.40	27.87	82.69
6. ClaimRank [SPEECHES, Exact]	31.15	51.60	76.71	26.50	30.95	68.20
7. ClaimRank + ELMo	41.67	57.73	90.47	24.98	43.73	83.12
8. ELMo (SPEECHES)	34.34			2.04	37.38	83.40
9. ELMo (SPEECHES, Similar)	36.22	62.74	89.41	36.07	40.73	81.89
10. ELMo (SPEECHES, Exact)	37.68	58.24	90.13	26.35	41.07	82.62

Table 4.2: Each of these models use a two layer feed-forward neural network classifier in order to predict the check-worthiness of an input claim. The feature inputs to this classifier are either a concatenation of the features in the ClaimRank model or a mean-pooling of the four layers of an ELMo model fine-tuned to the train set. All the models are cross validated, one at a time, against the four debates in the DEBATES dataset as was done in Section 3.2. One debate at a time is held out as the test set, one is held out as the validation set, and the last two are used for training. The datasets in parentheses are added into the train set for each of the four folds, and the datasets in brackets–rows 5, 6–are used to pre-train the classifier before fitting to the DEBATES dataset.

Chapter 5

Check-Worthy Claims and Wikipedia Dataset

As we have mentioned in previous chapters, there are several limitations with existing claim-detection datasets: they are small–CLEF-2018 has less than 10,000 claims [Atanasova et al., 2018], they are imbalanced–only 16.5% of claims in the Debate dataset are check-worthy (Table 3.2), and their subject matter comes from only the political sphere.

This chapter focuses on our development of and experimentation on a novel dataset for the claim-detection problem. We use sentences and citations within Wikipedia articles to define check-worthiness. The upcoming sections will cover our motivation for using Wikipedia as source data (Section 5.1), the dataset creation process (Section 5.2), an overall description of (Section 5.3) the dataset, the methodology (Section 5.4) behind our initial experiments (Section 5.5), and our analysis of the preliminary results (Section 5.6).

5.1 Motivation

The major push for fact-checking has primarily been motivated by the effects of 'fake news' on politics [Allcott and Gentzkow, 2017]. However, verifying the validity of a claim via evidence is a universal task, and providing appropriate sources to support

an argument or factual claim often constitutes finding evidence. Wikipedia's reported goal is to be a comprehensive source of information across all spheres of knowledge[1]. By virtue of being anonymously and collaboratively edited, Wikipedia's credibility is often questioned[2]. A backed-up claim is more credible than an unsubstantiated one. To increase credibility, Wikipedia requires that all content that is being or likely will be challenged to have a reliable source that can be checked by a visitor to the site[3]. In addition, Wikipedia requires that these sources appear in inline citations. This allows us to use the presence of citations to determine the check-worthiness of a sentence. Therefore, we use Wikipedia articles to create our claim-detection dataset.

Similarly to other claim-detection task setups, we consider each sentence in a Wikipedia article to be a claim. The presence of a citation means the claim is check-worthy, and the absence of a citation means the claim is not-check-worthy. On any given Wikipedia page, we can view sentences with inline citations as check-worthy claims and ones without inline citations as not-check-worthy claims. An example of a Wikipedia page with check-worthy claims highlighted is in Figure 5-1 [4][5].

Some sentences are marked with an inline [missing citation], which means that an editor decided this claim is likely to be challenged and thus check-worthy, but it has not yet been verified. An example of a claim with a [missing citation] tag is in Figure 5-2 [6].

Although all the articles on Wikipedia are held to an equally high standard, not all articles currently meet them. Wikipedia maintains and updates a list of articles of high quality articles, which they label as Good or Featured[7][8]. Of particular note is these articles have passed the highest standards of verifiability. Thus, the set of Good and Featured articles should provide us with the highest quality labels. The example in Figure 5-1 is taken from a Featured article.

[1] https://en.wikipedia.org/wiki/Wikipedia:Purpose
[2] https://en.wikipedia.org/wiki/Reliability_of_Wikipedia
[3] https://en.wikipedia.org/wiki/Wikipedia:Verifiability
[4] https://en.wikipedia.org/wiki/Oliver_Wendell_Holmes_Sr.
[5] https://en.wikipedia.org/wiki/Wikipedia:Featured_articles
[6] https://tools.wmflabs.org/citationhunt/
[7] https://en.wikipedia.org/wiki/Wikipedia:Good_articles/all
[8] https://en.wikipedia.org/wiki/Wikipedia:Featured_articles

Poetry [edit]

Holmes is one of the Fireside Poets, together with William Cullen Bryant, Henry Wadsworth Longfellow, James Russell Lowell, and John Greenleaf Whittier.[130] These poets—whose writing was characterized as family-friendly and conventional—were among the first Americans to build substantial popularity in Europe. Holmes in particular believed poetry had "the power of transfiguring the experiences and shows of life into an aspect which comes from the imagination and kindles that of others".[33]

Due to his immense popularity during his lifetime, Holmes was often called upon to produce commemorative poetry for specific occasions, including memorials, anniversaries and birthdays. Referring to this demand for his attention, he once wrote that he was "a florist in verse, and what *would* people say / If I came to a banquet without my bouquet?"[131] As critic Hyatt Waggoner noted, however, "very little ... survives the occasions that produced it".[132] Holmes became known as a poet who expressed the benefits of loyalty and trust at serious gatherings, as well as one who showed wit at festivities and celebrations. Edwin Percy Whipple for one considered Holmes to be "a poet of sentiment and passion. Those who know him only as a comic lyricist, as the libellous laureate of chirping follow and presumptuous egotism, would be surprised at the clear sweetness and skylark thrill of his serious and sentimental compositions".[133]

Figure 5-1: An example from the Wikipedia page of Oliver Wendell Holmes Sr. The highlighted parts are the claims with inline citations and the check-worthy claims. The page for Oliver Wendell Holmes Sr. is randomly chosen from the list of featured articles.

Citation Hunt

The Wikipedia snippet below is not backed by a reliable source. Can you find one?

Click **I got this!** to go to Wikipedia and fix the snippet, or **Next!** to see another one. Check Introduction to referencing with VisualEditor if you have questions. Good luck!

In page Artificial neural network:

> Models' "capacity" property roughly corresponds to their ability to model any given function. It is related to the amount of information that can be stored in the network and to the notion of complexity.[*citation needed*]

I got this! Next!

Search for a topic.

Figure 5-2: Example of a claim with a missing citation. The image is taken from web app Citation Hunt, which randomly shows Wikipedia claims which are missing citations.

5.2 Creation

Our data extraction pipeline can be split up into four steps: 1. Prepare a list of articles to extract claims from, 2. For a given article, extract the relevant text from the articles HTML source code, 3. Tokenize the extracted text into sentences, and 4. Label the tokenized sentences as check-worthy or not-check-worthy based on the presence of inline citations. In creating our dataset, we used the Urllib Python library to access Wikipedia articles HTML source codes, the BeautifulSoup4 Python library to parse the HTML, the Wikipedia Python library as a backup method to get the HTML files from Wikimedia, and the NLTK library to parse the articles into sentences. A diagram showing the steps is in Figure 5-3.

5.2.1 Creating the list of titles

To get a list of all Good and Featured articles, we first use Urllib to extract the HTML from `https://en.wikipedia.org/wiki/Wikipedia:Featured_articles` and `https://en.wikipedia.org/wiki/Wikipedia:Good_articles/all`. Both lists are consistently being updated as articles get higher in quality, so keeping an up to date dataset would require updating the list of Good and Featured articles. We extracted our title list on April 22, 2019. Because, both pages divide up articles using HTML tags, we use BeautifulSoup4 to extract the article names.

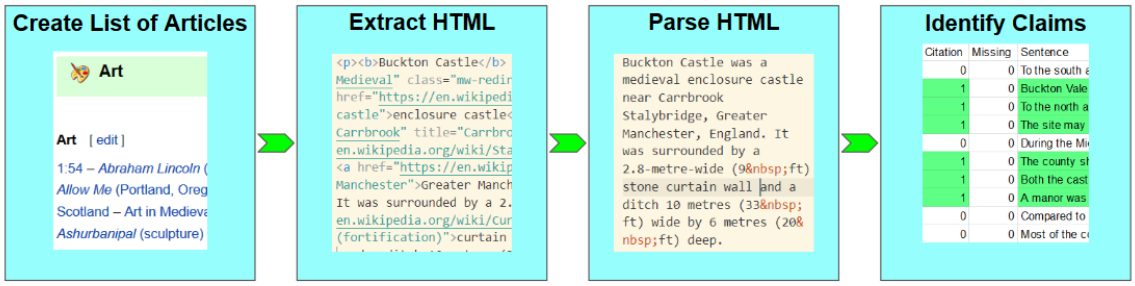


Figure 5-3: A diagram of the four step process in creating our Wikipedia dataset. This diagram shows the process to create the dataset entry for the article shown in Figure 5-8.

5.2.2 HTML extraction

Wikimedia stores information about every article in Wikipedia. Thus, using the Wikipedia library, we can easily and quickly access an article's text using just the article's title. However, Wikimedia does not consider inline citations as part of the article text, so we cannot use the Wikipedia library extract text features. Instead, we developed code to parse the HTML source code. Wikipedia articles' urls all follow the convention of https://en.wikipedia.org/wiki/ + [title_name]; thus we can use Urllib to extract the source HTML for any given article title. Using Urllib instead to extract HTML via web parsing is considerably faster than using the Wikipedia library to get the HTML from the Wikimedia database. However, we still need to use the Wikipedia library as a backup process for the very few circumstances in which Urllib is unable to return a page.

5.2.3 HTML parsing

To get the content of a Wikipedia article from its HTMLs source code, it is generally sufficient to extract all text that is found within paragraph blocks because Wikipedia places almost all its text content within blocks. An example is shown in Figure 5-4.

Wikipedia style dictates that the introduction of an article is not required to have citations as the introduction often contain claims that are more general than the main body of the paragraph, and some of the claims in the introduction will be further expanded upon later[9]. Thus, sometimes citations are omitted in the introduction to avoid redundant citations. Thus, for future work, it is worth considering not including claims made in the introduction as part of the dataset as sometimes check-worthy claims are not marked. However, this decision is left up to the discretion of the editor, so we included the introductions as part of our dataset because there are articles–especially introductions for complex topics–where the introduction does include inline citations[10].

[9] `https://en.wikipedia.org/wiki/Wikipedia:Man.ual_of_Style/Lead_section#Citations`
[10] `https://en.wikipedia.org/wiki/Pi` is one such example.

```
495   <p>Holmes is one of the <a href="https://en.wikipedia.org/wiki/
      Fireside_Poets" class="mw-redirect" title="Fireside Poets">Fireside Poets</a>,
       together with <a href="https://en.wikipedia.org/wiki/William_Cullen_Bryant"
      title="William Cullen Bryant">William Cullen Bryant</a>, Henry Wadsworth
      Longfellow, James Russell Lowell, and <a href="https://en.wikipedia.org/wiki/
      John_Greenleaf_Whittier" title="John Greenleaf Whittier">John Greenleaf
      Whittier</a>.<sup id="cite_ref-130" class="reference"><a
      href="#cite_note-130">[130]</a></sup>
496   These poets—whose writing was characterized as family-friendly and
497   conventional—were among the first Americans to build substantial
498   popularity in Europe. Holmes in particular believed poetry had "the
499   power of transfiguring the experiences and shows of life into an aspect
500   which comes from the imagination and kindles that of others".<sup
      id="cite_ref-Small,_37_33-1" class="reference"><a href="#cite_note-Small,
      _37-33">[33]</a></sup>
501   </p><p>Due to his immense popularity during his lifetime, Holmes was often
```

Figure 5-4: An example of the HTML source code for the example shown in Figure 5-1. This particular example is parsed and turned into an example in our dataset as shown in Table 5.1.

Citation	Missing Citation	Text	Title
1	0	Holmes is one of the Fireside Poets, together with William Cullen Bryant, Henry Wadsworth Longfellow, James Russell Lowell, and John Greenleaf Whittier.	Oliver Wendell Holmes Sr.
0	0	These poets whose writing was characterized as family-friendly and conventional were among the first Americans to build substantial popularity in Europe.	Oliver Wendell Holmes Sr.

Table 5.1: Two example claims that are created from the HTML shown in Figure 5-4.

5.2.4 Sentence splitting and claim labeling

Most inline citations occur at the end of sentences (see Figure 5-1), but the NLTK sentence tokenizer splits sentences based upon punctuation. Because of this, when we do not do additional processing, the sentence tokenizer considers end of sentence citations as part the next sentence. We notice that citations uniquely are brackets surrounding either a natural number of 'citation needed,' so we can use regular expressions to move our citation to before the end punctuation of a sentence. Moving the citations this way eliminated the above issue caused by running the NLTK sentence tokenizer without preprocessing the text.

When we split the paragraphs into sentences, we can easily label the sentences by using the same regular expressions to check for the presence of citation brackets

inside a sentence token. Then, after we label a sentences as either a check-worthy or a not-check-worthy claim, we remove the citation brackets and add that claim to our dataset. For every claim, we keep track of its corresponding article, the actual text of the claim, whether the claim had an inline citation, whether the claim was marked as needing a citation. An example of the end result of our extraction pipeline can be seen in Table 5.1.

5.3 Description

Our dataset can be subdivided up into categories based upon the categories that Wikipedia uses to divide its Good and Featured article [11] [12]. A list of all fifteen categories and examples articles for each of the categories can be found in Table 5.2.

A full description of our dataset in terms of number of articles, number of sentences, number of sentences with citations, and number of sentences that were marked as missing a citation can be found in Table 5.3. The table shows the quality of the verifiability of Good and Featured Wikipedia articles as less than only 0.1 percent were marked as needing a citation. In comparison to existing datasets, our dataset is much larger. Each of the categories on their own has more sentences in them than the CLEF-2018 dataset. In addition, our dataset is quite balanced, which satisfies one of goals for creating a new dataset. A histogram of the number of words per sentence across all the articles is shown in Figure 5-5, and a histogram of the number of sentences per article across all the articles is shown in Figure 5-6. Histograms of the number of words per sentence for all the sentences in each of the fifteen categories can be found in Section A.1. Histograms of the number of sentences per article for all the articles in each of the fifteen categories can be found in Section A.2.

Taking a deeper look into the number of sentences per article in Table 5.4, we can see that with the exception of Geography and Places articles being longer and Engineering and Technology articles being shorter, most of the categories have similar

[11] https://en.wikipedia.org/wiki/Wikipedia:Good_articles/all
[12] https://en.wikipedia.org/wiki/Wikipedia:Featured_articles

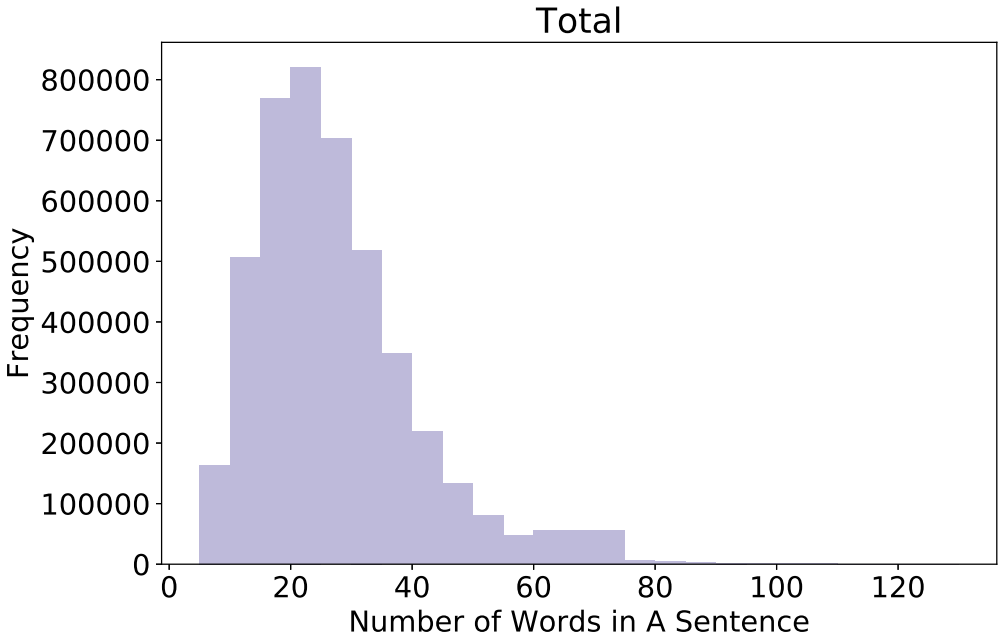


Figure 5-5: This is a histogram of the number of words per sentence across all 4,415,275 sentences.

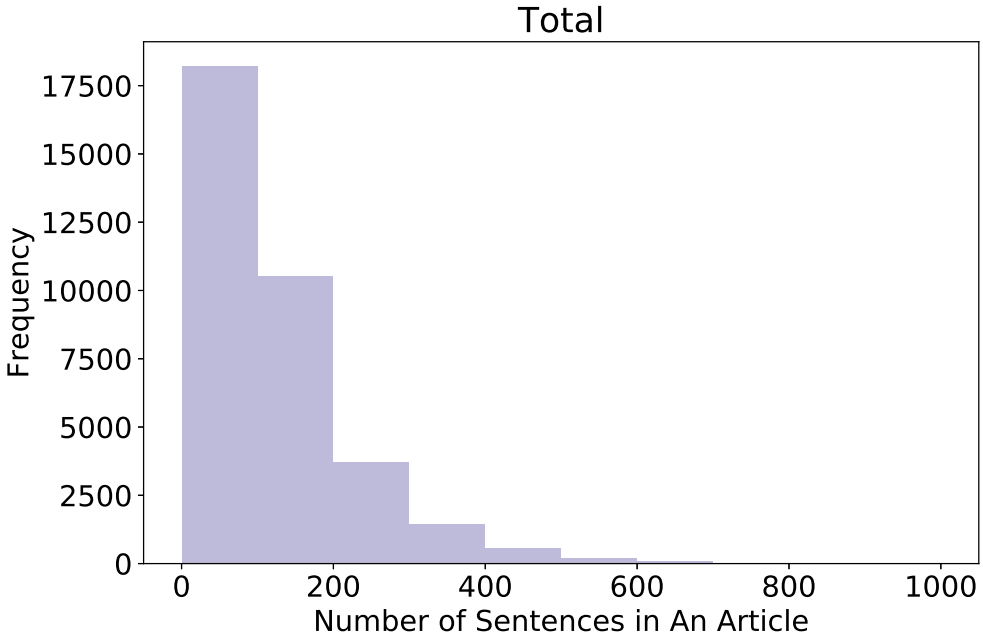


Figure 5-6: This is a histogram of the number of sentences per article across all 34,786 articles.

Category	Examples
Agriculture, Food, and Drink	Beer in North Korea, Cabbage, Maple syrup
Art and Architecture	Cloud Gate, Heian Palace, Scottish art in the nineteenth century
Engineering and Technology	Apollo 11, Renewable energy in Scotland, Apple TV
Geography and Places	River Brue, Appalachian Trail, Japan
History	Han dynasty, Great Stink, Octavia Hill
Language and Literature	International Phonetic Alphabet, Spider-Man, Poetry
Mathematics	Euclidean algorithm, Pi, Addition
Media and Drama	Fight Club, Sesame Street, Batman in film
Music	Kind of Blue, The Wood Nymph, The Royal Opera
Natural Sciences	Diamond, Protein, California sea lion
Philosophy and Religion	Isaac, Stoicism, Thor
Social Sciences and Society	Algorithmic bias, Royal National College for the Blind, Icelanders
Sports and Recreation	Brian Urlacher, Cy Young, Stan Musial
Video Games	Bastion (video game), Final Fantasy X, Mario Kart: Double Dash
Warfare	Battle of Salamis, Lapland War, Henry Wrigley

Table 5.2: Three example articles for each of the fifteen categories.

makeups in terms of article length. This provides evidence towards the feasibility of a generalizable claim-detection model. Table 5.3 reinforces this claim as most of the claims are quite balanced in terms of check-worthy versus not-check-worthy claims. The exception to this is the Mathematics class, which leans more towards the check-worthy class. A possible explanation would be that Mathematics articles have a heavy emphasis on proofs, and it is redundant to cite a textbook for every single line of a proof. However, even the Mathematics class is more balanced than any of the existing claim-detection datasets.

Table 5.5 shows the complexity of the articles in a category. Longer sentences are likely to be more complicated than shorter ones. As Table 5.5 shows, all the categories have almost identical makeups in terms of words per sentence. The most noticeable difference between the categories is the difference in the max number of words in a sentence. However, on closer inspection, we see that these outlier sentences are specific edge cases that our parser was not able to catch. Examples include when a sentence ends not in a block. Unfortunately, in some of these cases, the correct behavior is to not assume the sentence is over because the other half of the sentence is in the next block, but in some other cases, the correct behavior is to assume the sentence is over, because the end of the sentence occurs

Category	Articles	Sentences	Citation	No Citation	% Citation	Missing	% Missing
Agriculture, Food, and Drink	236	24,850	13,183	11,667	53.05	30	0.12
Art and Architecture	1,276	143,871	76,426	67,445	53.12	94	0.07
Engineering and Technology	2,621	243,657	117,801	125,856	48.35	198	0.08
Geography and Places	1,216	224,614	116,066	108,548	51.67	242	0.11
History	2,256	355,563	193,196	162,367	54.34	325	0.09
Language and Literature	1,277	174,132	86,674	87,458	49.77	112	0.06
Mathematics	85	14,153	4,817	9,336	34.04	10	0.07
Media and Drama	4,281	529,870	247,973	281,897	46.80	385	0.07
Music	4,125	458,402	270,125	188,277	58.93	317	0.07
Natural Sciences	4,329	530,984	264,141	266,843	49.75	365	0.07
Philosophy and Religion	664	98,068	53,064	45,004	54.11	109	0.11
Social Sciences and Society	2,313	363,557	203,864	159,693	56.07	344	0.09
Sports and Recreation	4,059	519,743	297,802	221,941	57.30	429	0.08
Video Games	1,492	188,233	100,834	87,399	53.57	112	0.06
Warfare	4,556	545,578	243,754	301,824	44.68	278	0.05
Total	34,786	4,415,275	2,289,720	2,125,555	51.86	3350	0.08

Table 5.3: Statistics of the number of articles, sentences, citations, and missing citations across the different categories.

in some other HTML block. Other errors that occur include ones where our HTML parser includes embedded style codes as part of the sentence.

The other side of the spectrum, where we have sentences of length 0 or 1 are generally sentence tokenization artifacts, where we have leftover floating quotation marks or empty spaces between two paragraph blocks. We decided to remove these outlier cases by constricting our training and testing dataset to only incorporate sentences between 5 and 128 words inclusive. The lower end was chosen because the 1 percentile mark hovers between 5 and 6 percent. The upper end was chosen because that is the input size of the $\text{BERT}_{\text{base}}$ uncased model.

5.4 Method

In creating our dataset, we want to make sure that all the claims for a single article would be in the same partition. Table 5.4 also shows that the average number of sentences per article is very high and relatively consistent between categories so splitting our claims into training and testing sets based upon article titles would not shortchange either of them.

We first randomized all the articles and then took 20 percent of the articles to be the test set and left the other 80 percent of the articles for the training step. We did

Category	Mean	Std	Min	25%	50%	75%	Max
Agriculture, Food, and Drink	105.29	81.69	23	48	82	131	479
Art and Architecture	112.75	89.78	18	53	83	142	587
Engineering and Technology	92.96	82.24	10	38	65	116	647
Geography and Places	184.71	127.37	11	83	159	254	813
History	157.61	128.62	13	66	115	207	943
Language and Literature	136.36	94.48	15	69	109	177	707
Mathematics	166.51	111.95	14	79	134	245	547
Media and Drama	123.77	86.72	19	65	93	156	809
Music	111.13	85.38	15	56	86	138	868
Natural Sciences	122.66	92.20	10	58	91	156	685
Philosophy and Religion	147.68	113.82	11	63	112	203	698
Social Sciences and Society	157.17	119.28	13	72	119	206	857
Sports and Recreation	128.04	101.92	10	55	100	169	810
Video Games	126.16	72.29	18	74	109	161	533
Warfare	119.75	99.31	10	57	87	145	951
Total	126.92	100.03	10	59	95	162	951

Table 5.4: Statistics of the number of sentences per article across different categories. For each category and all the categories combined, we show the mean, the standard deviation, the 0 percentile (Min), 25 percentile, 50 percentile, 75 percentile, and 100 percentile (Max) of the number of sentences in a given article.

not purposefully ensure that the two datasets have the same category proportion as the full dataset, but as Table 5.6 shows, the test set and the training set had similar category proportions.

All our subsequent experiments were thus trained using our training set partition and evaluated on our testing set partition. Because there is no article overlap between the two datasets, high performing models need to be able to generalize to unseen topics.

Because the best performing models in our previous chapters were based around BERT, we chose to use BERT as the foundation of our baseline models. In particular, this means that our models are context-free, which should work well for Wikipedia claims because Wikipedia articles are not argumentative. Each sentence should stand on its own as a potentially verifiable fact.

Our model is similar to the one used in Section 3.2.2. We start with a $\text{BERT}_{\text{base}}$

Category	Mean	Std	Min	1%	10%	25%	50%	75%	90%	99%	Max
Agriculture, Food, and Drink	26.14	13.08	1	6	13	17	24	32	42	69	292
Art and Architecture	26.37	12.98	1	6	13	17	24	33	42	66	408
Engineering and Technology	25.56	15.62	1	5	12	17	24	32	40	63	704
Geography and Places	25.20	12.39	1	7	12	17	23	31	40	64	663
History	26.77	12.91	1	6	13	18	25	33	43	67	601
Language and Literature	27.70	14.20	1	5	13	18	25	35	45	73	469
Mathematics	30.14	20.07	1	5	13	18	26	37	51	99	547
Media and Drama	26.23	13.04	0	5	12	17	24	33	43	67	299
Music	26.60	13.34	0	4	12	17	25	33	43	68	725
Natural Sciences	24.54	12.14	1	6	12	16	22	30	39	62	1,021
Philosophy and Religion	27.12	14.00	1	6	13	18	25	34	44	72	289
Social Sciences and Society	27.41	14.20	0	6	13	18	25	34	44	72	950
Sports and Recreation	24.82	11.64	0	6	12	17	23	31	39	61	615
Video Games	26.39	12.59	1	7	13	18	24	33	42	65	850
Warfare	26.18	12.18	1	6	13	18	24	33	42	63	365
Total	26.05	13.03	0	6	12	17	24	32	42	66	1,021

Table 5.5: Statistics of the number of words per sentence across different categories. For each category and all the categories combined, we show the mean, the standard deviation, the 0 percentile (Min), 1 percentile, 10 percentile, 25 percentile, 50 percentile, 75 percentile, 99 percentile, and 100 percentile (Max) of the number of words in a given sentence as determined by the NLTK word tokenizer.

uncased model that was initialized with pre-trained weights[13]. This model takes as input our tokenized claim that is prepended with a special [CLS] token. We then use BERT as a sentence level encoder and take the output node that is positionally the same as our [CLS] token as the input to a feed-forward neural network classifier, which gives our claim a predicted label of check-worthy or not-check-worthy. After some trial and error with different hyperparmeters and optimizers, we found that the best ones were the default recommended BERT settings.

5.5 Experimental Settings

We focused on two different sets of experiments. Our first experiment fine tuned our BERT-based model to the training set. For the other experiment, we attempted first tuning on the Debate datasets to hopefully teach the model with respect to the claim-detection task before fine tuning the model on our Wikipedia training set. We only trained our models for 5 epochs because our datasets is quite large and the BERT

[13] `https://github.com/huggingface/pytorch-pretrained-BERT`

Category	Article (Train)	Sentence (Train)	Article (Test)	Sentence (Test)
Agriculture, Food, and Drink	184	18,956	52	5,751
Art and Architecture	1,032	114,708	244	28,381
Engineering and Technology	2,082	193,032	539	48,479
Geography and Places	988	182,642	228	41,284
History	1,815	283,259	441	70,816
Language and Literature	1,030	140,990	247	31,789
Mathematics	75	12,526	10	1,441
Media and Drama	3,463	426,436	818	99,989
Music	3,268	359,260	857	94,885
Natural Sciences	3,458	421,954	871	107,181
Philosophy and Religion	521	79,304	143	18,199
Social Sciences and Society	1,875	288,952	438	72,345
Sports and Recreation	3,247	413,985	812	103,305
Video Games	1,181	149,030	311	38,498
Warfare	3,612	427,669	944	115,057
Total	27,831	3,512,703	6,955	877,400

Table 5.6: The number of articles and sentences in our train and test sets. We only include sentences with word counts between 5 and 128 words.

paper used a small number of epochs for their reported experiments [Devlin et al., 2018].

For both sets of experiments, we check-pointed our model after 1, 3, and 5 epochs in order to investigate how our BERT model changed over time. Again, our classifier minimizes a binary cross-entropy loss function.

5.6 Results and Analysis

A full set of results can be found in Table 5.7. We broke down the results of our best model, row 2, into categories in Table 5.6.

Comparing rows 2-4 with rows 5-7 in Table 5.7 shows that pre-training on the Debate dataset is generally an ineffective strategy. This makes sense because the Wikipedia dataset is just so much larger in comparison, and the based pre-trained model has already shown itself to be very effective on a variety of NLP tasks [Devlin et al., 2018].

Comparing rows 2-7 with row 1 shows that there are word and sentence level characteristics that are unique to the claim-detection task. Even though our pre-trained model was trained using a prediction task on a different Wikipedia-based

dataset, the initial model was not nearly as good as our tuned model. Thus, our model must have been learning additional linguistic and syntactical features that are specifically helpful for claim-detection.

An example output of our best model can be see in Figure 5-7. On inspection of this example, it appears that our model seems to making reasonable estimations as to what is and is not-check-worthy. Table 5.7 shows that our models perform better on the positive check-worthy class than on the negative not-check-worthy class.

Model	**AvgP**	**F1**	**F1 Neg.**	**F1 Pos.**	**R Precision**	**Accuracy**
1. $\text{BERT}_{\text{base}}$ uncased	53.00	32.57	63.59	1.54	54.72	46.84
2. BERT (1 epoch)	76.21	67.95	65.62	70.28	69.57	68.12
3. BERT (3 epochs)	75.28	67.30	64.91	69.70	68.91	67.48
4. BERT (5 epochs)	72.49	66.14	64.00	68.28	67.82	66.28
5. BERT (pre-trained, 1 epoch)	76.20	67.97	65.43	70.52	69.57	68.17
6. BERT (pre-trained, 3 epochs)	75.29	67.31	64.62	69.99	68.93	67.53
7. BERT (pre-trained, 5 epochs)	72.47	66.15	63.98	68.31	67.79	66.28

Table 5.7: Each of the models are initialized using the $\text{BERT}_{\text{base}}$ uncased model. The hyper-parameters are set to the default BERT settings. We train two models, one is fitted just to the Wikipedia dataset and the other is fitted first to the Debate dataset then the Wikipedia dataset, for five epochs. The number of epochs in parentheses gives which checkpoint model used to evaluate the test set.

One hypothesis for why it is that our model has no sense of where in an article it is, so it cannot pick up on the fact that introduction sections generally speaking have much fewer citations than other sections. We can see this when we look at what our model does on introduction sections such as in 5-8 [14]. As our example shows, each of the highlighted claims ends up being elaborated on in more detail later on in the article, and each of those elaborations had citations. Because article introductions almost all have no citations, all of these arguably check-worthy claims are counted against our model. For future work, we plan on not incorporating the introduction sections because they easily confound our results.

[14] https://en.wikipedia.org/wiki/Buckton_Castle

Poetry [edit]

Holmes is one of the Fireside Poets, together with William Cullen Bryant, Henry Wadsworth Longfellow, James Russell Lowell, and John Greenleaf Whittier.[130] These poets—whose writing was characterized as family-friendly and conventional—were among the first Americans to build substantial popularity in Europe. Holmes in particular believed poetry had "the power of transfiguring the experiences and shows of life into an aspect which comes from the imagination and kindles that of others".[33]

Due to his immense popularity during his lifetime, Holmes was often called upon to produce commemorative poetry for specific occasions, including memorials, anniversaries and birthdays. Referring to this demand for his attention, he once wrote that he was "a florist in verse, and what *would* people say / If I came to a banquet without my bouquet?"[131] As critic Hyatt Waggoner noted, however, "very little ... survives the occasions that produced it".[132] Holmes became known as a poet who expressed the benefits of loyalty and trust at serious gatherings, as well as one who showed wit at festivities and celebrations. Edwin Percy Whipple for one considered Holmes to be "a poet of sentiment and passion. Those who know him only as a comic lyricist, as the libellous laureate of chirping follow and presumptuous egotism, would be surprised at the clear sweetness and skylark thrill of his serious and sentimental compositions".[133]

Figure 5-7: An example output of our best performing model from Table 5.7 on the example section shown in Figure 5-1 from the Featured article on Oliver Wendell Holmes Sr. The highlighted sentences are the ones that our model predicted a check-worthy label.

5.6.1 Trends in loss function

We graphed out the training and evaluation loss over time for both our experimental setups in Figure 5-9. As we can see, our model seems to overfit almost immediately. This may be because our model was initialized on a Wikipedia-based dataset, so it is already fine-tuned to have contextual word embeddings that well represent our words as the model has already seen all our input sentences, just in different contexts. Thus, our classifier and BERT model only needed a small shift to be adjust the classifier from predicting the next following sentence to claim-detection.

5.7 Category specific performance

When we break down our model performance into categories, as is shown in Table 5.8, we can see that our model's performance varies depending on category type. Interestingly, it appears that our model does worse on science and technology-related categories than humanities-related ones. This implies that characteristics of check-worthy claims are category dependent. This could be because check-worthiness standards probably vary depending on the field, and the difference is the most apparent when comparing humanities to sciences as they are commonly referred to as opposite

Buckton Castle was a medieval enclosure castle near Carrbrook in Stalybridge, Greater Manchester, England. It was surrounded by a 2.8-metre-wide (9 ft) stone curtain wall and a ditch 10 metres (33 ft) wide by 6 metres (20 ft) deep. Buckton is one of the earliest stone castles in North West England and only survives as buried remains overgrown with heather and peat. It was most likely built and demolished in the 12th century. The earliest surviving record of the site dates from 1360, by which time it was lying derelict. The few finds retrieved during archaeological investigations indicate that Buckton Castle may not have been completed.

In the 16th century, the site may have been used as a beacon for the Pilgrimage of Grace. During the 18th century, the castle was of interest to treasure hunters following rumours that gold and silver had been discovered at Buckton. The site was used as an anti-aircraft decoy site during the Second World War. Between 1996 and 2010, Buckton Castle was investigated by archaeologists as part of the Tameside Archaeology Survey, first by the University of Manchester Archaeological Unit then the University of Salford's Centre for Applied Archaeology. The project involved community archaeology, and more than 60 volunteers took part. The castle, close to the Buckton Vale Quarry, is a Scheduled Ancient Monument.

Buckton was a small highland enclosure castle with a 2.8-metre-thick (9 ft) sandstone curtain wall; nothing survives above ground.[33]

The dearth of artefacts recovered from Buckton Castle and the lack of finely finished stonework indicate that the site was never finished, but the re-cutting of the ditch suggests either an extended period of occupancy or abandonment followed by repairs to the fortifications.[17]

the excavations throughout. More than 60 volunteers were involved in the excavations between 2007 and 2010, including people from the Tameside Archaeological Society, the South Trafford Archaeological Group, the South Manchester Archaeological Research Team, and students from several universities.[32]

castle.[22] Since 1924, the castle has been designated as a Scheduled Ancient Monument,[23] which is intended to protect important archaeological sites from change;[24] this was probably to protect the castle from Buckton Vale Quarry as it expanded.[25] During the Second World

Figure 5-8: (Top) An example introduction section that has been labeled by our model. Buckton Castle, the subject of this passage, has a Featured article about it. As you can see, the original introduction contains zero citations in it, but our model specifies four sentences as check-worthy. Indeed, when these four claims are elaborated on later, which are shown underneath the introduction, they all need to have inline citations.

Category	AvgP	F1	F1 Neg.	F1 Pos.	R Precision	Accuracy
BERT (1 epoch)	76.21	67.95	65.62	70.28	69.57	68.12
1. Music	85.29	72.74	65.64	79.84	77.93	74.59
2. Media and Drama	80.67	75.84	75.89	75.78	74.05	75.84
3. Sports and Recreation	79.97	68.22	60.65	75.79	73.28	70.02
4. Language and Literature	78.69	70.47	69.52	71.43	71.16	70.50
5. Video Games	78.28	69.28	66.93	71.63	71.31	69.46
6. Social Sciences and Society	76.01	64.76	58.02	71.49	69.77	66.04
7. Art and Architecture	73.50	64.51	61.54	67.49	67.00	64.76
8. History	72.97	63.49	59.28	67.71	67.43	63.98
9. Philosophy and Religion	72.19	63.85	59.90	67.80	67.03	64.28
10. Agriculture, Food, and Drink	71.25	64.29	60.17	68.42	66.61	64.77
11. Engineering and Technology	70.92	66.57	67.55	65.59	65.76	66.60
12. Geography and Places	69.06	62.69	62.38	63.01	64.42	62.69
13. Natural Sciences	68.69	64.32	64.12	64.52	64.42	64.32
14. Mathematics	67.54	68.81	78.16	59.46	62.83	71.62
15. Warfare	66.03	64.13	68.96	59.30	61.43	64.78

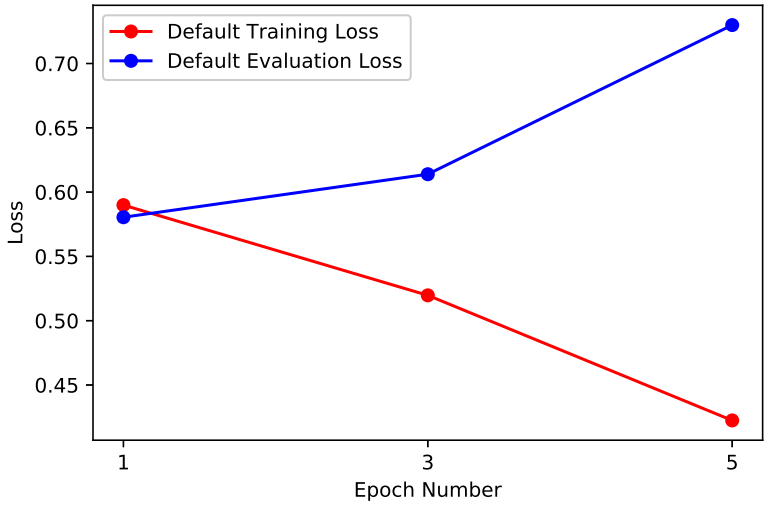
Table 5.8: This table shows the performance of the best performing model in Table 5.7–the default $\text{BERT}_{\text{base}}$ uncased model tuned on our dataset for one epoch–on individual categories. Our table begins with the overall performance of the model before sorting the categories based on highest average precision.

fields.

5.8 Summary

In summary, we motivated and introduced our Wikipedia claim-detection dataset in this chapter. Our experiments with tuning BERT based classifiers on our dataset and analyzing its outputs, we have shown a context-free model like ours is unable to pick up on the nuances of article introduction sections and properly adjust its check-worthiness criteria to account for the general lack of citations in article introductions.

We have also shown through experiments that there are claim-detection specific features that differ from category to category. Just looking at word and sentence distributions among articles, it may seem that different categories are quite similar. But, breaking our model performance into category-specific performance, we can see that they are actually quite different.



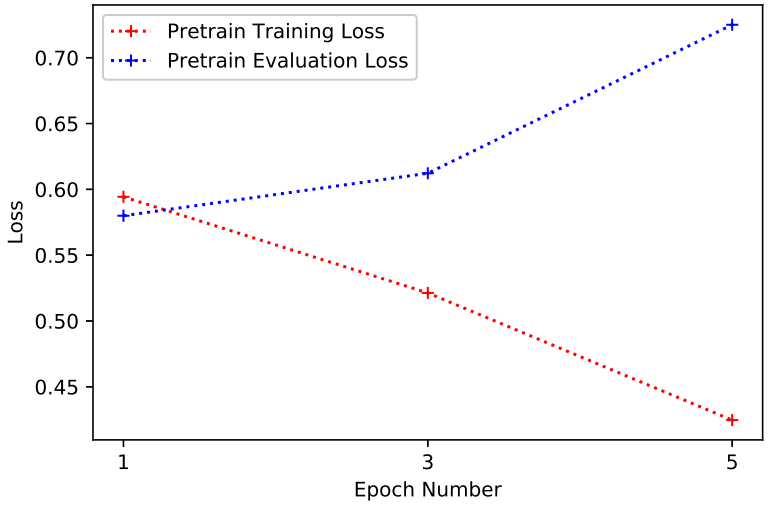


Figure 5-9: Graphs show the loss of the models in Table 5.7 during train epochs. The graphs compare the training and evaluation loss for the $\text{BERT}_{\text{base}}$ uncased model tuned on our Wikipedia dataset, rows 2–4 in Table 5.7 (top), and for the $\text{BERT}_{\text{base}}$ uncased model that first tuned on the Debate dataset before being tuned on the Wikipedia dataset, rows 5–7 in Table 5.7.

Chapter 6

Check-Worthy Claims and CLEF-2019 Dataset

As mentioned in Section 2.1, six of the eight sources from the Debate dataset that we used for training our models in Chapter 3 were released as part of the CLEF-2018 CheckThat! Lab on Automatic Identification and Verification of Political Claims Check-worthiness challenge [Nakov et al., 2018; Atanasova et al., 2018]. CLEF-2019 released the same challenge this year, but added in an additional 16 annotated sources. Nine of these sources were released as part of the training set for participants in addition to the sources from CLEF-2018. The final seven of these sources were used to test the models sent in by the participants in the check-worthiness challenge. We helped with the manual annotation process in creating the CLEF-2019 check-worthiness dataset, and this chapter will describe the dataset 6.1 as well as the annotation process 6.2.

6.1 Dataset Description

Our source transcripts were taken from FactCheck.org [1] One of the key points of FactCheck.org is that their articles and transcripts focus on claims that are false or

[1] `https://transcripts.factcheck.org/president-trump-paris-climate-accord/`. An example source.

misleading[2] In particular, this means that if they start fact-checking a claim and learn that the statement is both true and clear, then they will move on and the claim will not be labeled in the later published article.

Therefore, the claim-detection problem as laid out by the CLEF-2019 dataset is fundamentally different than the one we have been focusing on. Specifically, the CLEF-2019 dataset is based around a task which combines both fact-checking and claim-detection. When FactCheck.org annotates a claim in one of their articles, the claim is either false or misleading, meaning that models trained on FactCheck.org transcripts need to be able to identify both factuality and clarity. Not only that, we still run into the problem of not being certain that every false and every unclear claim in the source has been annotated.

A full description of the sources we annotated this year can be found in Table 6.1. The table shows that there are considerably fewer check-worthy claims (proportionally speaking) in this dataset than the Debate dataset described in Section 3.2. This is due to what we mentioned earlier about how FactCheck.org only annotates false and misleading claims. This claim-detection is much harder as it is more specific than the claim-detection task for ClaimRank, which helps to explain why the best submitted model for the CLEF-2019 Challenge has a mean average precision of 16.6 percent.[3]

6.2 Annotation

As mentioned in Section 6.1, we based all our annotations on published transcripts from FactCheck.org. The first step of converting the transcripts into individual sentence claims was already done for us. We helped with the annotation portion of the dataset creation. For a given source, we looked through the FactCheck.org transcript and looked for annotated sentences. We then found the corresponding sentence in our sentence split copy and marked it as a claim. Then, we check the sentences and

[2] `https://www.factcheck.org/our-process/`. Description of FactCheck.org's fact-checking process.

[3] `https://github.com/apepa/clef2019-factchecking-task1` Evaluation results for and the full annotated datsaet of the CLEF-2019 claim-detection task can be found here.

Claims	Sents	N Claims	% Claim	Type	Date (YMD)	Source
11	1,761	22	1.25	p	20180926	President Trumps U.N. Press Conference
11	1,250	14	1.12	p	20170216	President Trump Press Conference
9	173	19	10.98	s	20170601	President Trump on the Paris Climate Accord
8	407	7	1.72	s	20170315	Remarks by the President in Nashville & Tennessee
8	44	8	18.18	?	20181010	Medicare For All
5	71	9	12.68	s	20190108	Trump's Oval Office Pitch on Border Wall/Immigration
6	446	15	3.36	d	20181112	Trump-Pelosi-Schumer Scuffle
12	2,136	34	1.59	d	20160226	10th GOP debate
10	1,187	23	1.94	d	20160209	MSNBC democratic debate
8	1,388	10	0.72	d	20151219	3rd democratic debate
8	1,716	23	1.34	d	20160311	12th GOP debate
6	612	14	2.29	p	20181015	Trumps 60 Minutes Interview
13	520	16	3.08	s	20180131	Trump's State of the Union address
14	859	25	2.91	s	20190215	President Trumps National Emergency Remarks
14	504	9	1.79	s	20190205	State of the Union Address
8	1,480	13	0.88	d	20160129	7th GOP Debate
80	7,475	151	2.02			Training Set
71	7,079	110	1.55			Test Set
151	14,554	261	1.79			Total

Table 6.1: Statistics for only the portion of the CLEF-2019 dataset that is annotated this year. The Claims columns keep track of the number of claims listed by FactCheck.org, but N Claims is the number of claims we found that matched the claims listed by FactCheck.org. % Claim equals N Claims divided by the number of sentences. For Type, p means press conference, s means speech, and d means debate.

unmarked claims that are unrelated to the additional FactCheck.org commentary on the marked claim. The hardest part, which required our discretion, was to look at the surrounding claims for ones that were related but had not been originally annotated. An example of this can be seen in Figure 6-1[4]. As we can see in Table 6.1, there was only a single source, the 'Medicare for All' source, for which the claims we marked matched the FactCheck.org annotations exactly. Because the final step of checking to unmark annotated claims and mark unannotated claims requires our judgment call, all the source annotations were checked by at least one other person. In the preparation for the CLEF-2019 dataset, we both created and checked labels based upon the FactCheck.org annotations.

[4] `https://transcripts.factcheck.org/president-trump-paris-climate-accord/` Source for the upper image.

THE PRESIDENT: Thank you very much. (Applause.) Thank you. I would like to begin by addressing the terrorist attack in Manila. We're closely monitoring the situation, and I will continue to give updates if anything happens during this period of time. But it is really very sad as to what's going on throughout the world with terror. Our thoughts and our prayers are with all of those affected.

Before we discuss the Paris Accord, I'd like to begin with an update on our tremendous — absolutely tremendous — economic progress since Election Day on November 8th. The economy is starting to come back, and very, very rapidly. We've added $3.3 trillion in stock market value to our economy, and more than a million private sector jobs.

I have just returned from a trip overseas where we concluded nearly $350 billion of military and economic development for the United States, creating hundreds of thousands of jobs. It was a very, very successful trip, believe me. (Applause.) Thank you. Thank you.

TRUMP	Before we discuss the Paris Accord, I'd like to begin with an update on our tremendous — absolutely tremendous — economic progress since Election Day on November 8th.	1	1
TRUMP	The economy is starting to come back, and very, very rapidly.	1	1
TRUMP	We've added $3.3 trillion in stock market value to our economy, and more than a million private sector jobs.	1	1
TRUMP	I have just returned from a trip overseas where we concluded nearly $350 billion of military and economic development for the United States, creating hundreds of thousands of jobs.	1	1

Figure 6-1: Graphics of our annotation interface for annotating the 'President Trump on the Paris Climate Accord' press conference. The top picture is from FactCheck.org, and the bottom is our Google Drive annotation scheme. Notice that in the top picture only the first three sentences were highlighted and labeled as a fact-checked claim, but we labeled the fourth sentence as part of the same claim because the sentence is still about economic progress during Trump's presidency.

6.3 Summary

In summary, we have shown, through describing the annotation process of the CLEF-2019 check-worthiness challenge dataset, the limitations of using journalists as the gold standard from which to obtain claim-detection labels. This chapter focuses primarily on two issues. The first being that fact-checking organizations focus mostly on finding and publishing falsehoods, and so there is a dearth of labeled true and check-worthy claims. The second is that journalist annotations are not perfect, and so extra manual processing is needed in order to create a complete set of gold standard labels.

Chapter 7

Conclusion

7.1 Summary

In this thesis, we have shown the deficiencies in existing claim-detection datasets and have explored ways to combat the various problems via creating additional data. We compared and contrasted the Debate dataset (Chapter 3), a new Politifact-based dataset (Chapter 4), a new Wikipedia-based dataset (Chapter 5), and the CLEF-2019 dataset (Chapter 6). We showed how the creation process of a dataset could bias the dataset away from a correct claim-detecting task. We analyzed how using different word embeddings to create sentence vectors can affect a model to learn the nuances of check-worthy claims.

7.1.1 Debate

The Debate dataset is created based on fact-checking organization annotations, which, as we showed, creates issues regarding completeness of labelling and generalizability. We introduced a model based on a strong contextual word embedding called BERT that was able to beat the overall performance of previous models on the Debate dataset. Our experiments also revealed how spoken claims are often check-worthy due to the context in which they are found (see Chapter 3).

7.1.2 Politifact

Our experiments on Politifact dataset provided insight into the effectiveness of ELMo word embeddings for claim-detection. Our work on this dataset showed that there are some major differences between spoken and written check-worthy claims. In particular, the context around a spoken claim forms a cohesive unit that should be considered together whereas a written claim is often a quotation or statement of fact that stands on its own (see Chapter 4).

7.1.3 Wikipedia

Our creation of the Wikipedia dataset allowed us to explore how check-worthiness varies in contexts/domain outside of politics. Our experiments confirmed that there are definitely sentence and word level features that correspond to check-worthiness. While, the experiments also showed that these check-worthiness features can vary across domains (see Chapter 5).

7.1.4 CLEF-2019

Our work to annotate the dataset for the CLEF-2019 claim-detection task confirmed issues surrounding journalist-based datasets. The primary concern with this dataset, as was the concern with the Debate dataset, is its class imbalance. There are way more not-check-worthy claims in this dataset than check-worthy claims. This issue results in the models can be easily biased to the majority class (see Chapter 6).

7.2 Future Work

In this thesis, we aimed to present datasets and models to address the check-worthy claims detection task, which is the first step of an automated fact-checking system. We further plan to continue working on this task and present efficient and effective models for the task.

Current bottlenecks include the lack of quality data in different domains additional to politic domain, which we attempted to alleviate via our creation of Wikipedia-based dataset. Currently, our models on this dataset only consider context within a sentence without considering the context outside the input claim. While, we know that context is very important for spoken claims (see Chapters 3 and 4). Thus, we plan to incorporate the context outside the input claim in our experiments on this dataset. In addition, while, our dataset was able to alleviate some of the concerns regarding completeness and generalizability, there are still some lingering issues with our dataset, and some of the issue are related to the parsing shortcomings. Thus, we plan to clean up the dataset without using any hand-crafted rules to eliminate the claims that contain parsing artifacts.

In addition, we showed that how variance in domain can lead to a variance in model performance (see Chapter 5). As future work, it is worthwhile to consider splitting up train set into category/domain-based ones rather than integrating them all into a train set. This can help a model to pick up the domain specific features during training.

As mentioned earlier, our final goal is detecting check-worthy claims for an automatic fact-checking system. Thus, we plan to incorporate check-worthy module trained on our Wikipedia dataset–as it contains different domains–into a fact-checking system. The module will take an input document, label its claims based upon how much they need verification, and return a list of check-worthy claims–similar to Figure 5-7 in Chapter 5–to specify the input claims to the fact-checking system.

Appendix A

Wikipedia Statistics

Section A.1 contains histograms of the number of words per sentence across all the sentences in each of the fifteen categories and across all fifteen categories. All the histograms are very similar in shape.

Section A.2 contains histograms of the number of sentences per article across all the articles in each of the fifteen categories and across all fifteen categories. With the exception of the histogram for the category Geography and Places, all the other histograms have similar shapes.

A.1 Histograms of the Number of Words per Sentence

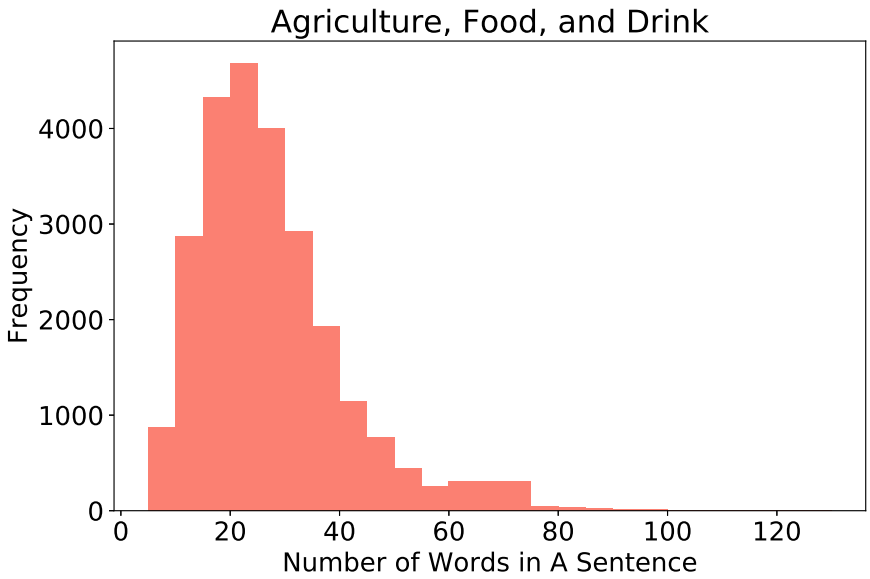


Figure A-1: This is a histogram of the number of words per sentence across all 24,850 sentences in the Agriculture, Food, and Drink category.

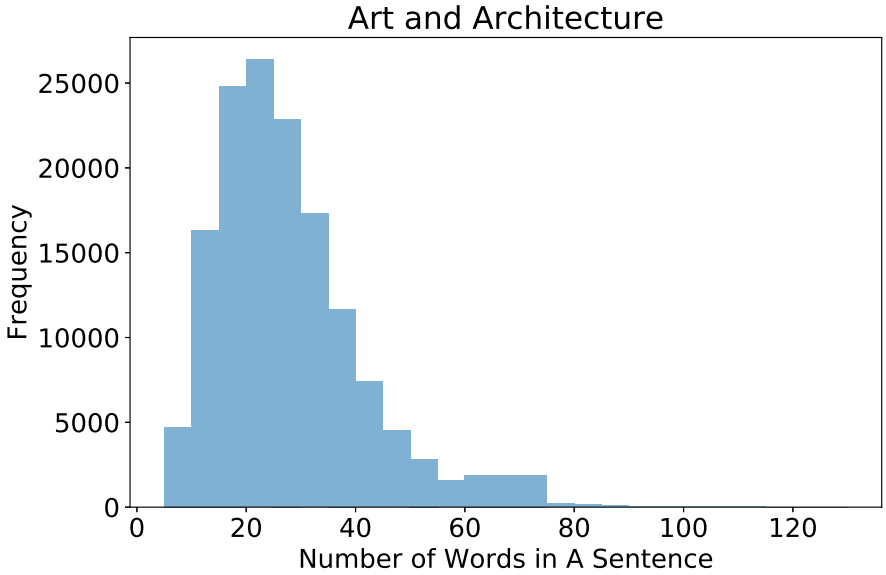


Figure A-2: This is a histogram of the number of words per sentence across all 143,871 sentences in the Art and Architecture category.

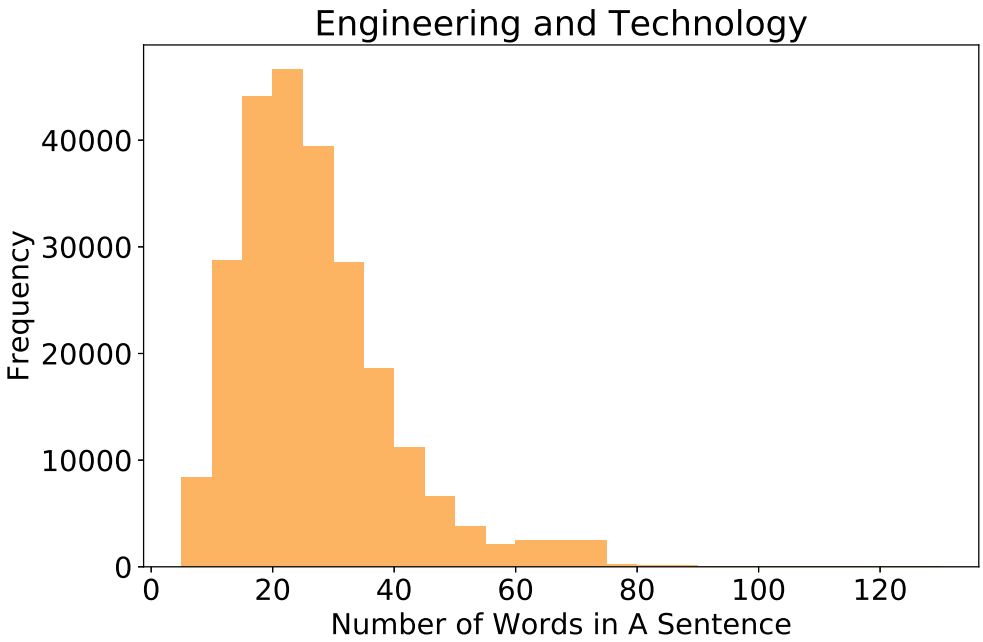


Figure A-3: This is a histogram of the number of words per sentence across all 243,657 sentences in the Engineering and Technology category.

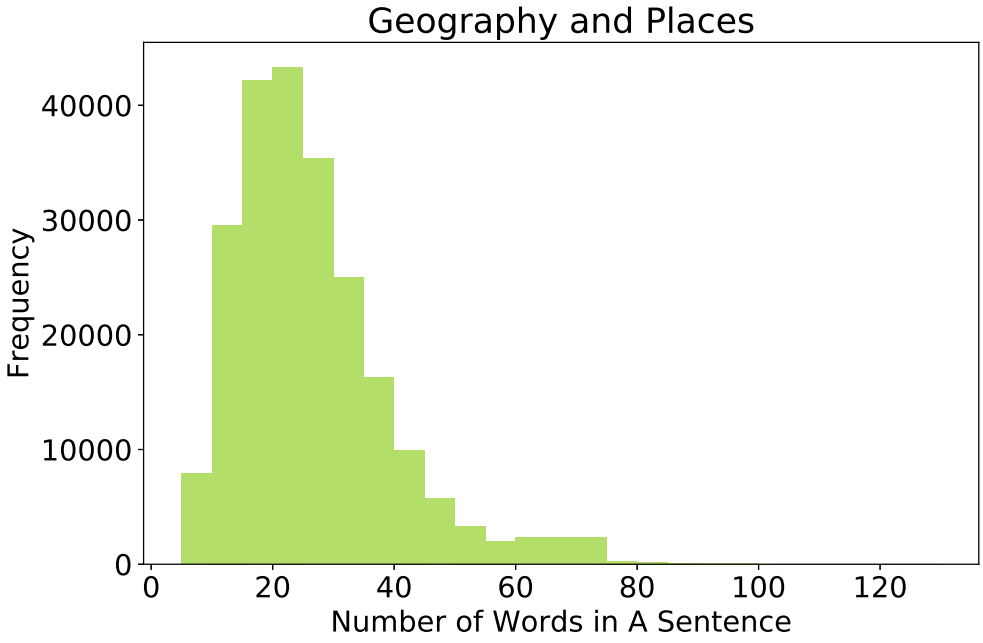


Figure A-4: This is a histogram of the number of words per sentence across all 224,614 sentences in the Geography and Places category.

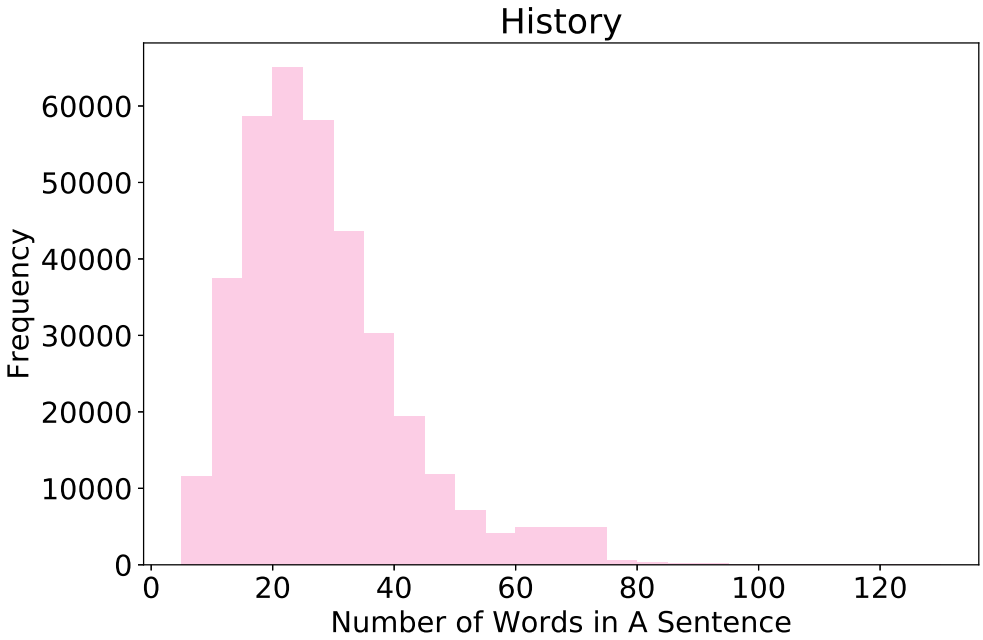


Figure A-5: This is a histogram of the number of words per sentence across all 355,563 sentences in the History category.

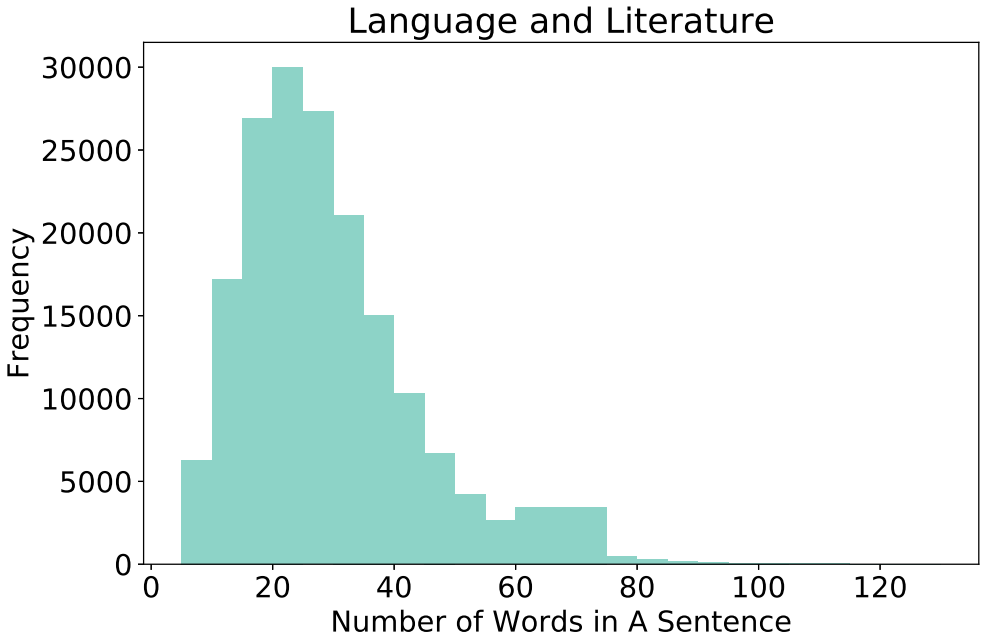


Figure A-6: This is a histogram of the number of words per sentence across all 174,132 sentences in the Language and Literature category.

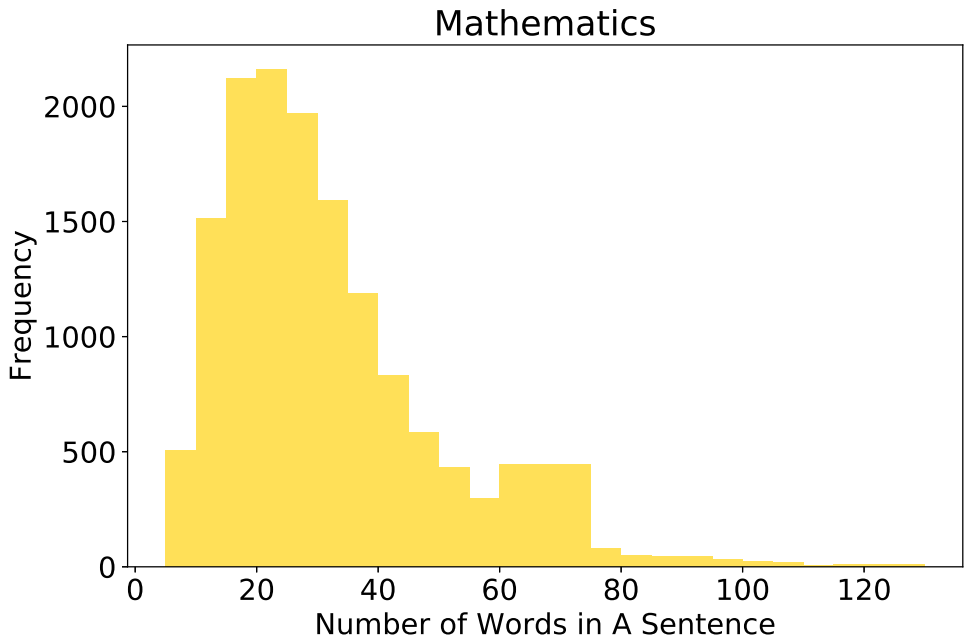


Figure A-7: This is a histogram of the number of words per sentence across all 14,153 sentences in the Mathematics category.

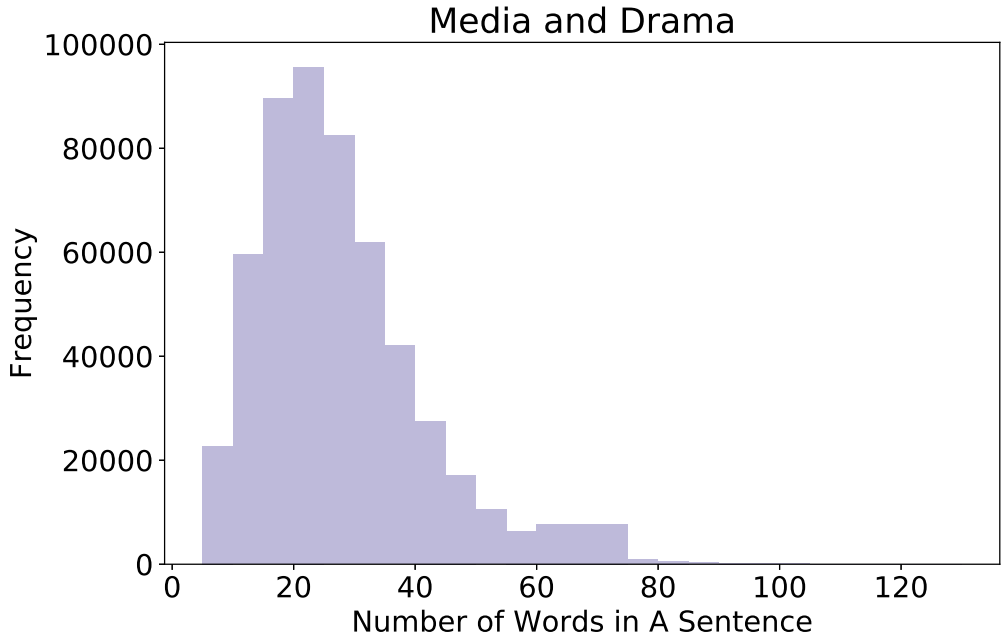


Figure A-8: This is a histogram of the number of words per sentence across all 529,870 sentences in the Media and Drama category.

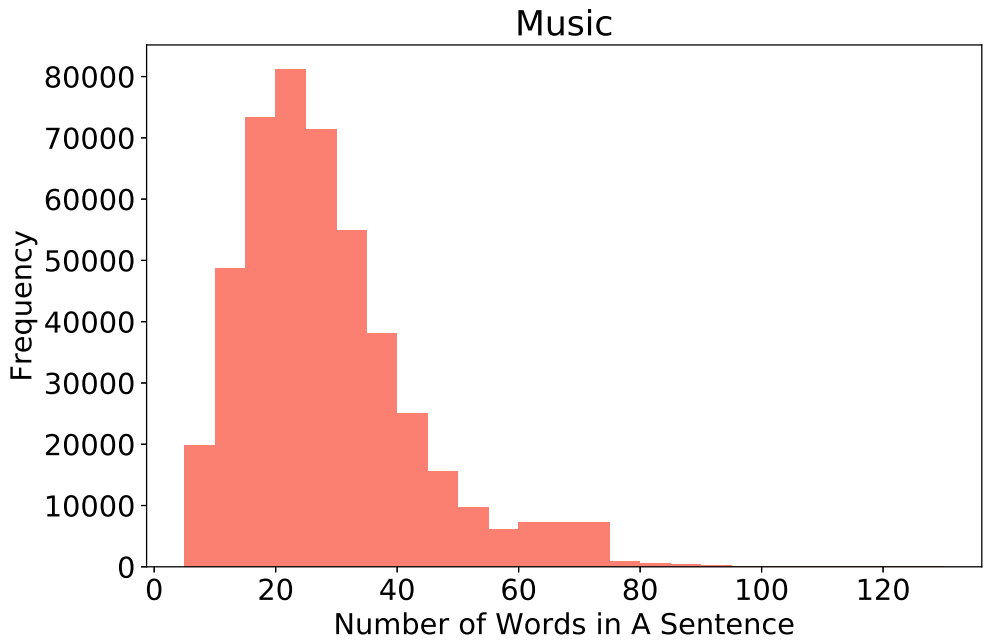


Figure A-9: This is a histogram of the number of words per sentence across all 458,402 sentences in the Music category.

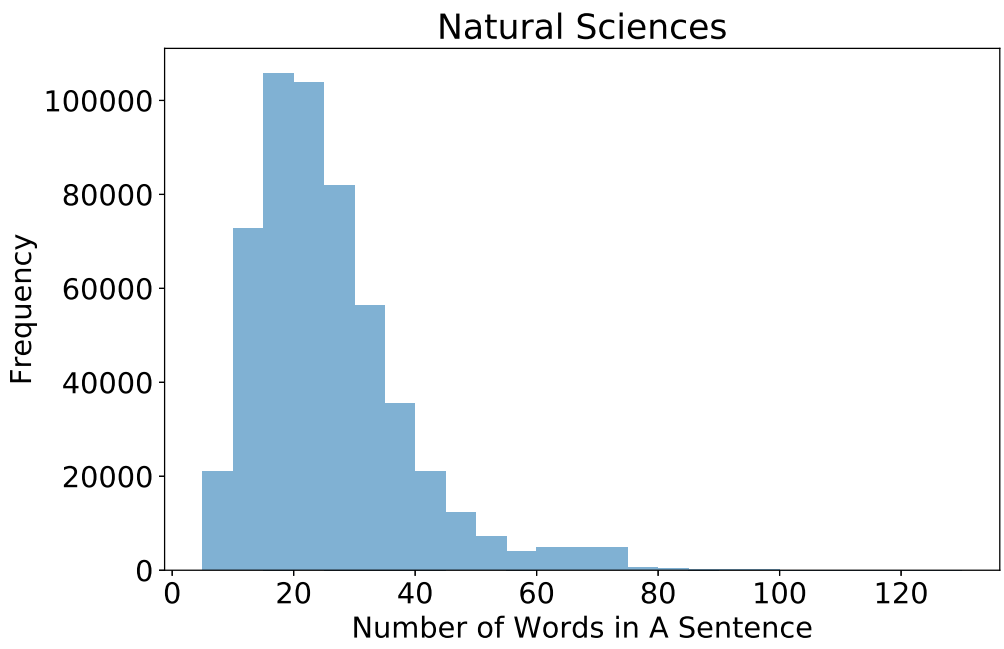


Figure A-10: This is a histogram of the number of words per sentence across all 530,984 sentences in the Natural Sciences category.

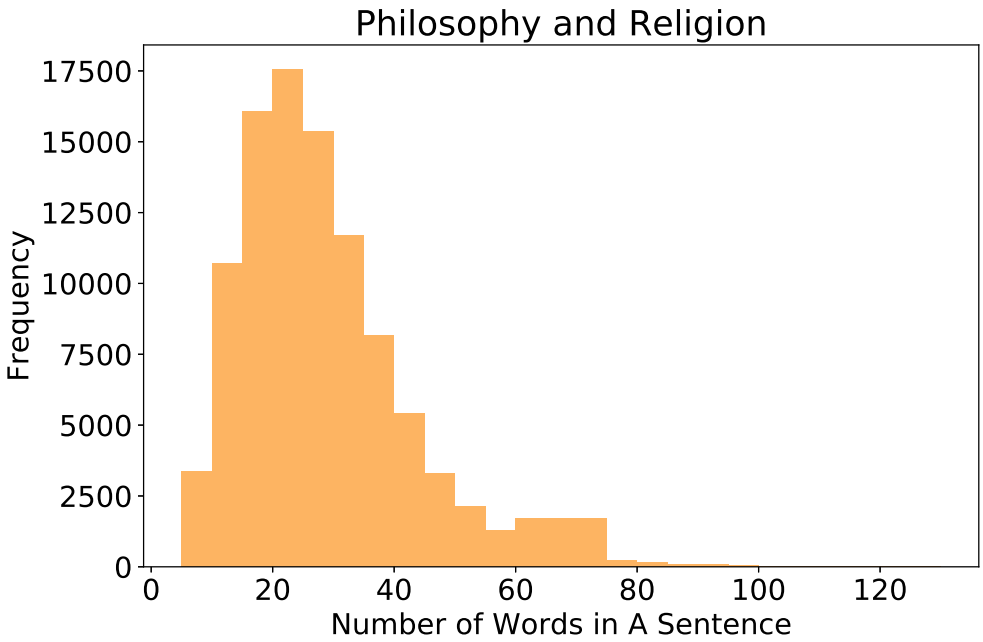


Figure A-11: This is a histogram of the number of words per sentence across all 98,068 sentences in the Philosophy and Religion category.

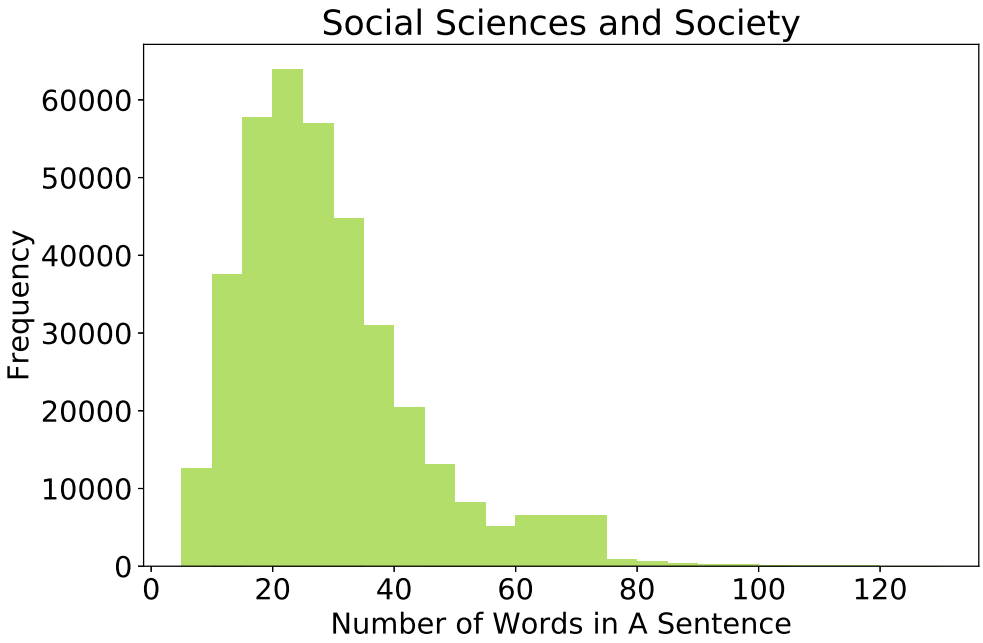


Figure A-12: This is a histogram of the number of words per sentence across all 363,557 sentences in the Social Sciences and Society category.

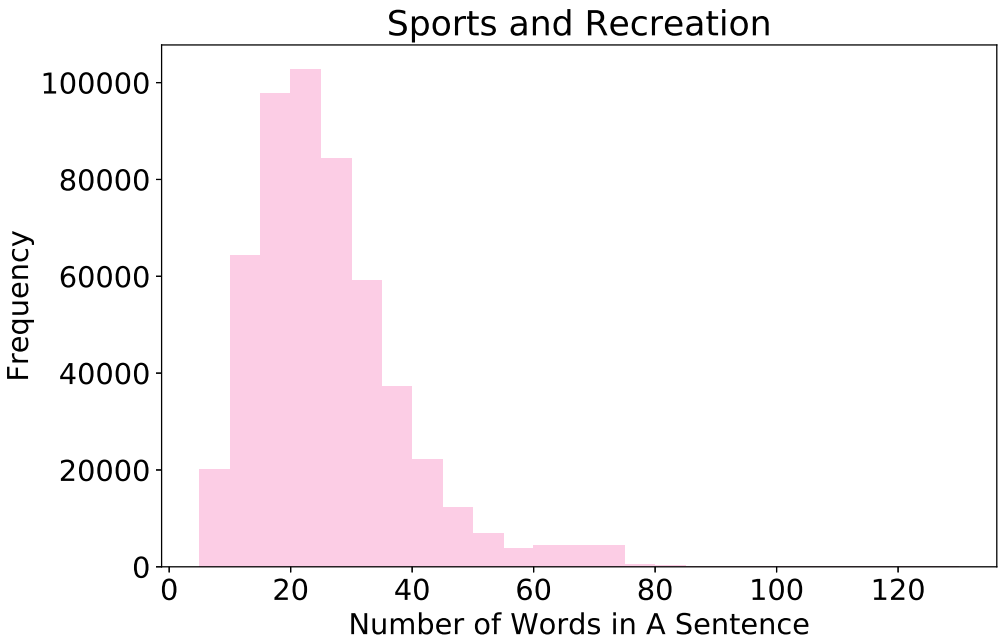


Figure A-13: This is a histogram of the number of words per sentence across all 519,743 sentences in the Sports and Recreation category.

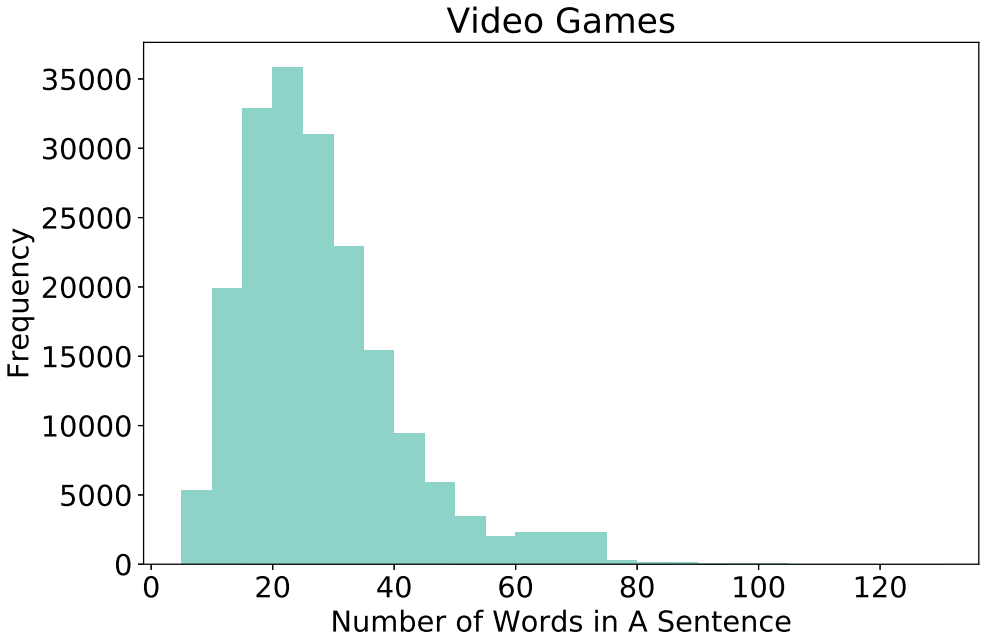


Figure A-14: This is a histogram of the number of words per sentence across all 188,233 sentences in the Video Games category.

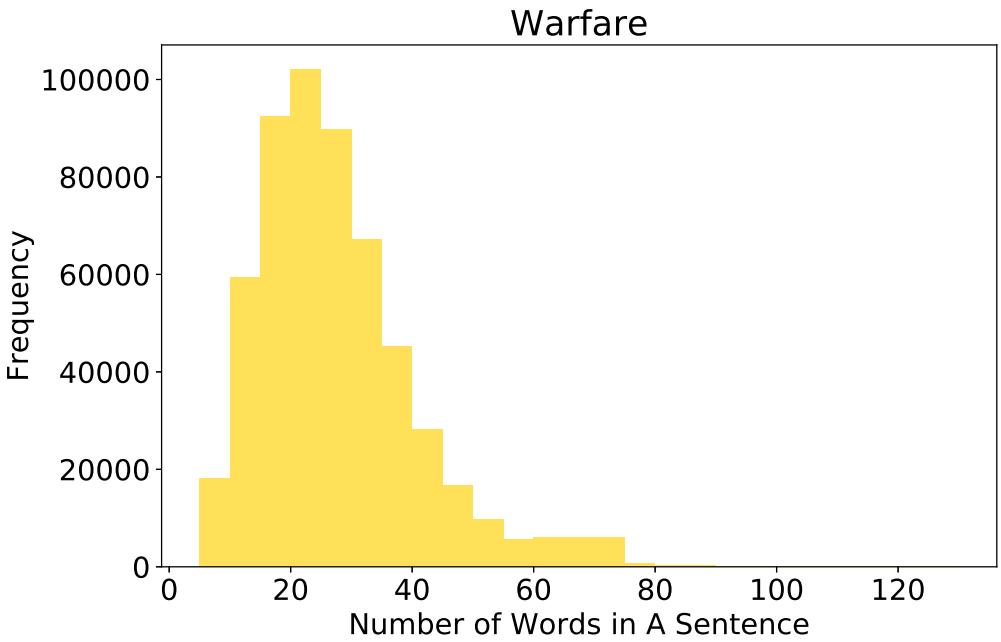


Figure A-15: This is a histogram of the number of words per sentence across all 545,578 sentences in the Warfare category.

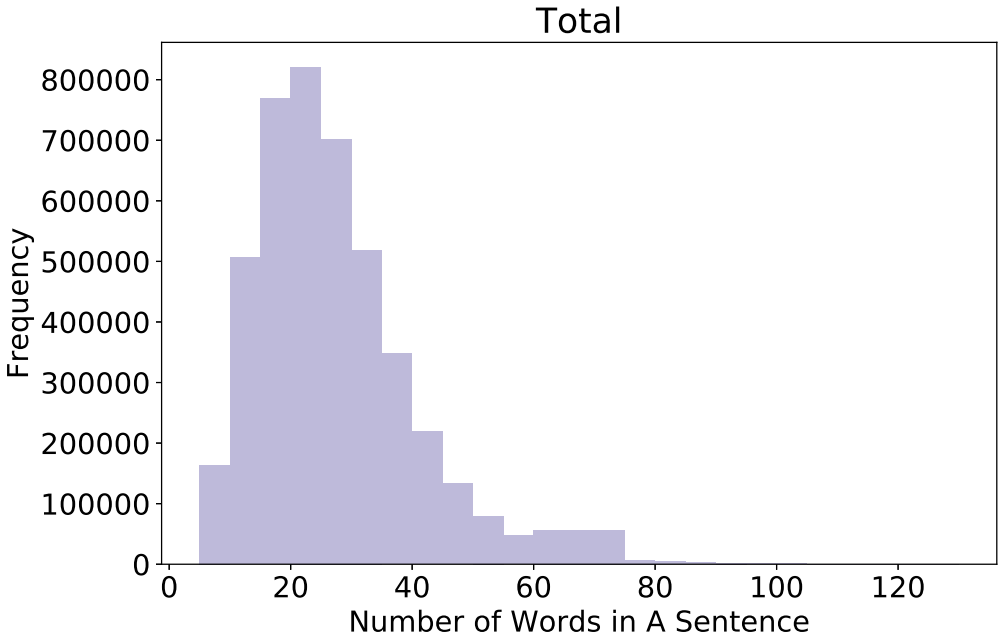


Figure A-16: This is a histogram of the number of words per sentence across all 4,415,275 sentences.

A.2 Histograms of the Number of Sentences per Article

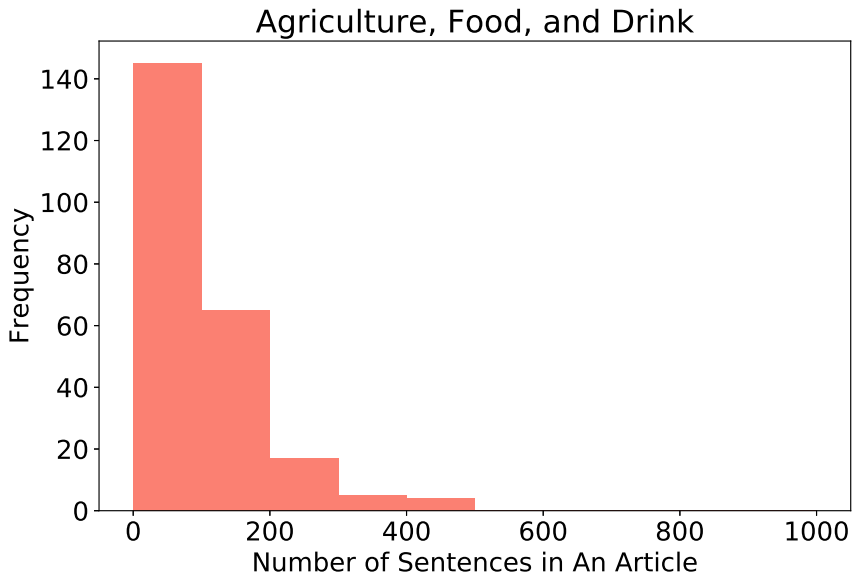


Figure A-17: This is a histogram of the number of sentences per article across all 236 articles in the Agriculture, Food, and Drink category.

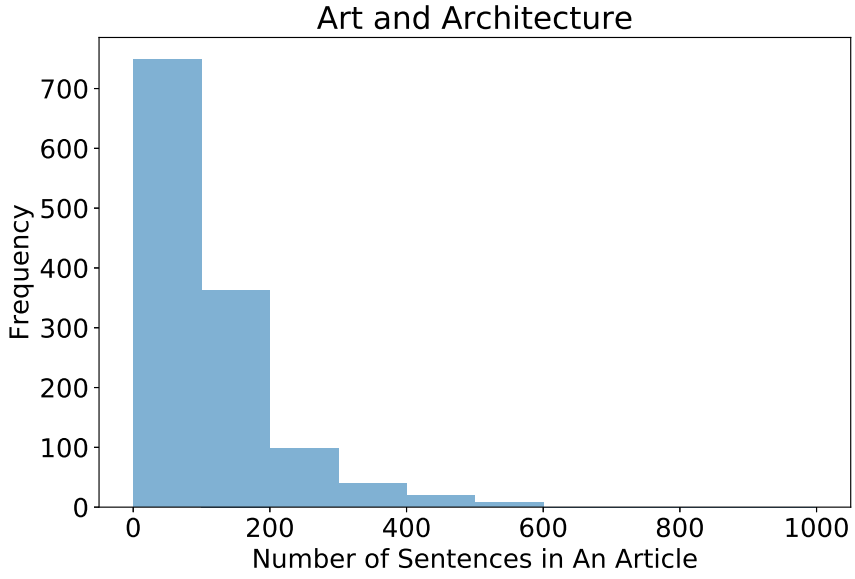


Figure A-18: This is a histogram of the number of sentences per article across all 1,276 articles in the Art and Architecture category.

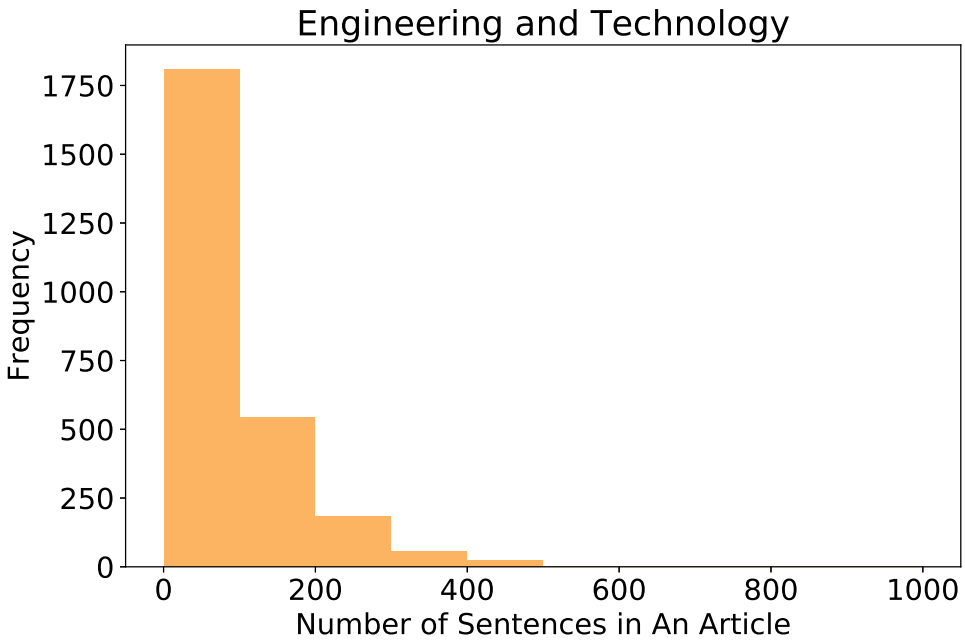


Figure A-19: This is a histogram of the number of sentences per article across all 2,621 articles in the Engineering and Technology category.

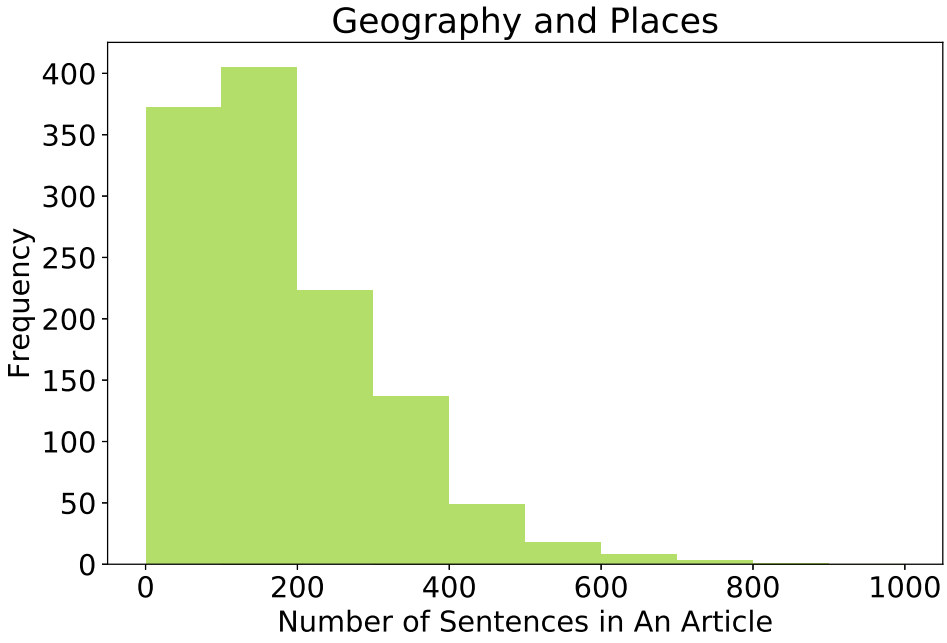


Figure A-20: This is a histogram of the number of sentences per article across all 1,216 articles in the Geography and Places category.

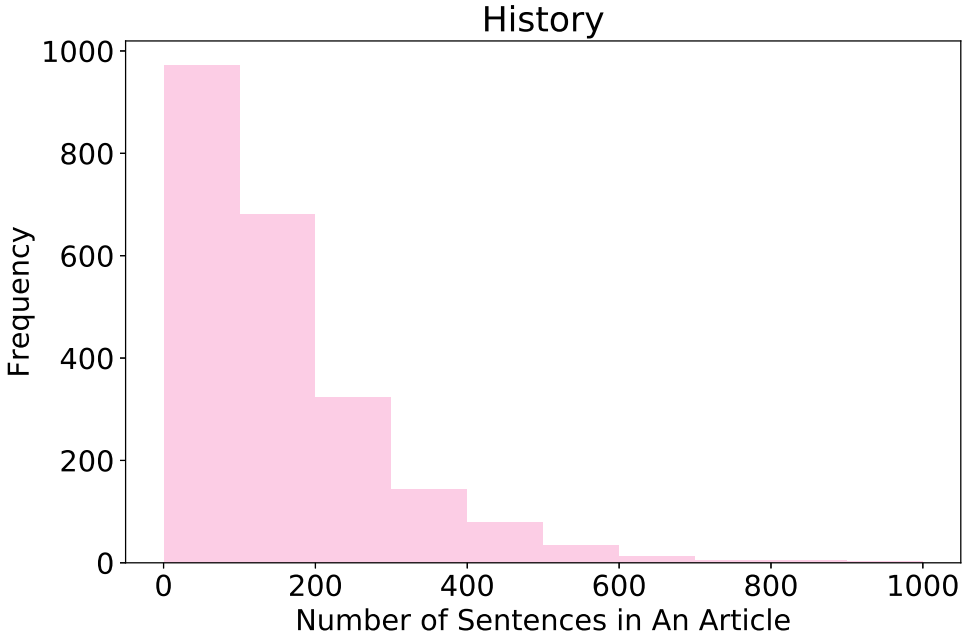


Figure A-21: This is a histogram of the number of sentences per article across all 2,256 articles in the History category.

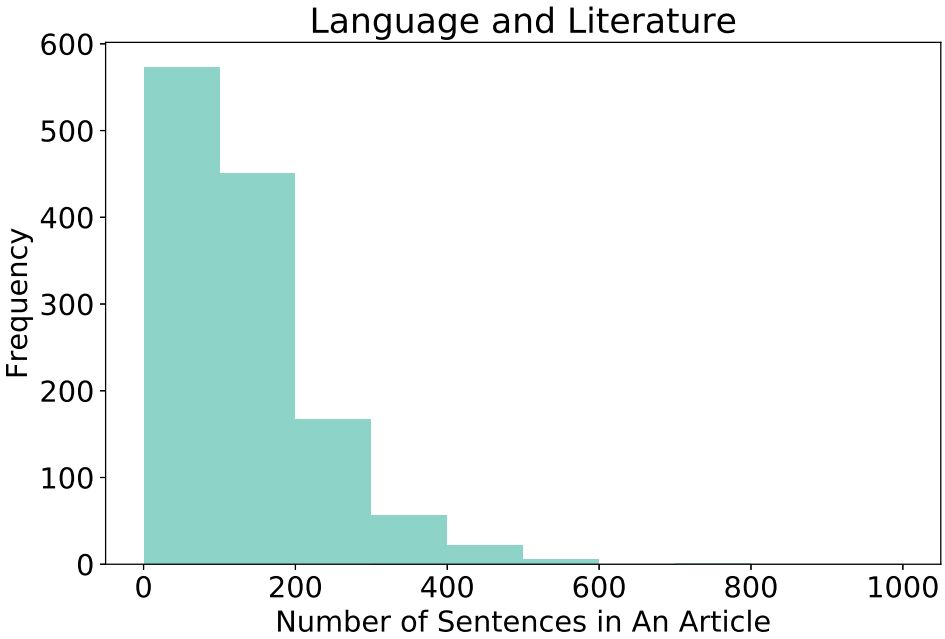


Figure A-22: This is a histogram of the number of sentences per article across all 1,277 articles in the Language and Literature category.

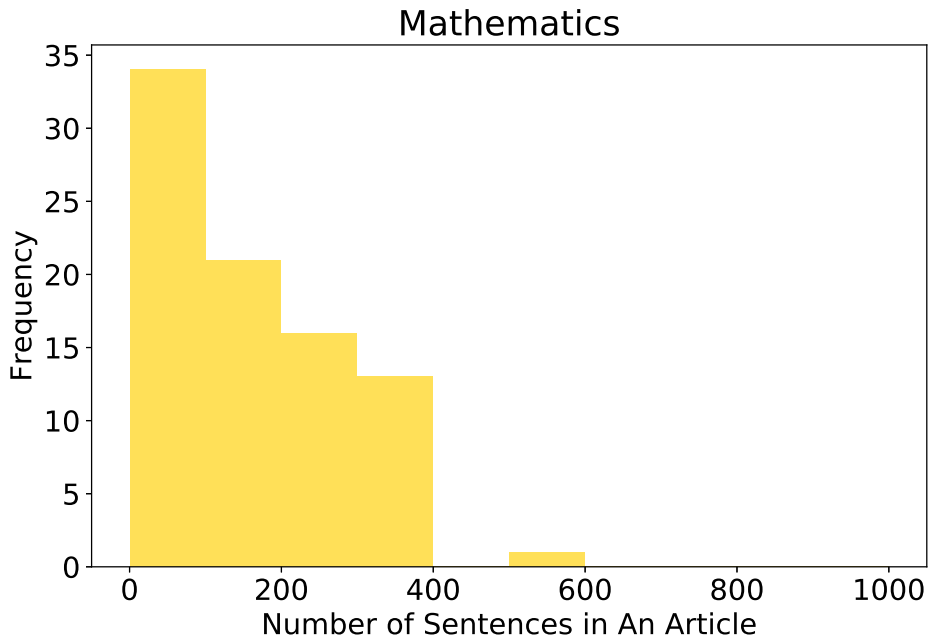


Figure A-23: This is a histogram of the number of sentences per article across all 85 articles in the Mathematics category.

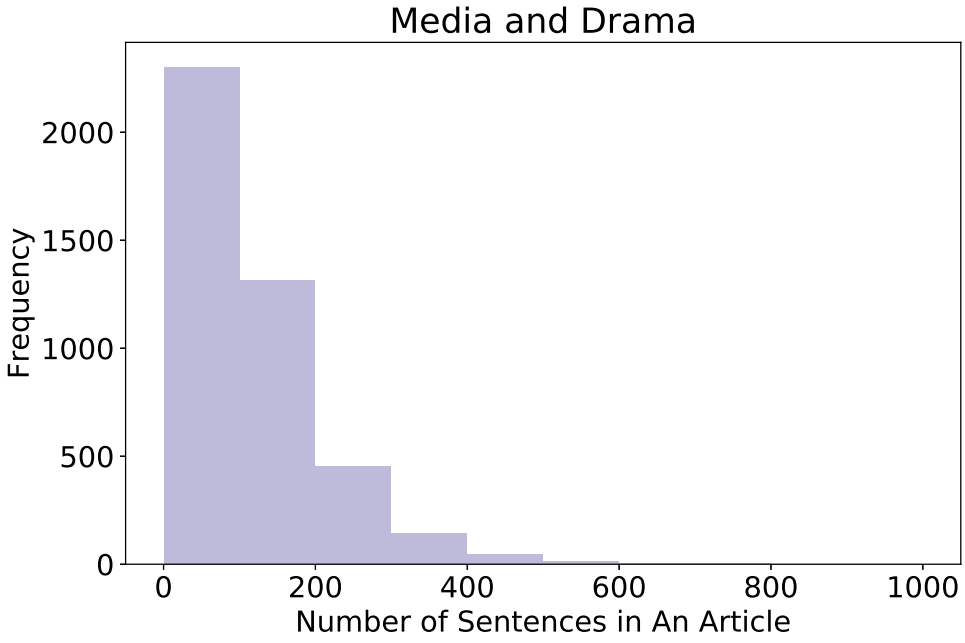


Figure A-24: This is a histogram of the number of sentences per article across all 4,281 articles in the Media and Drama category.

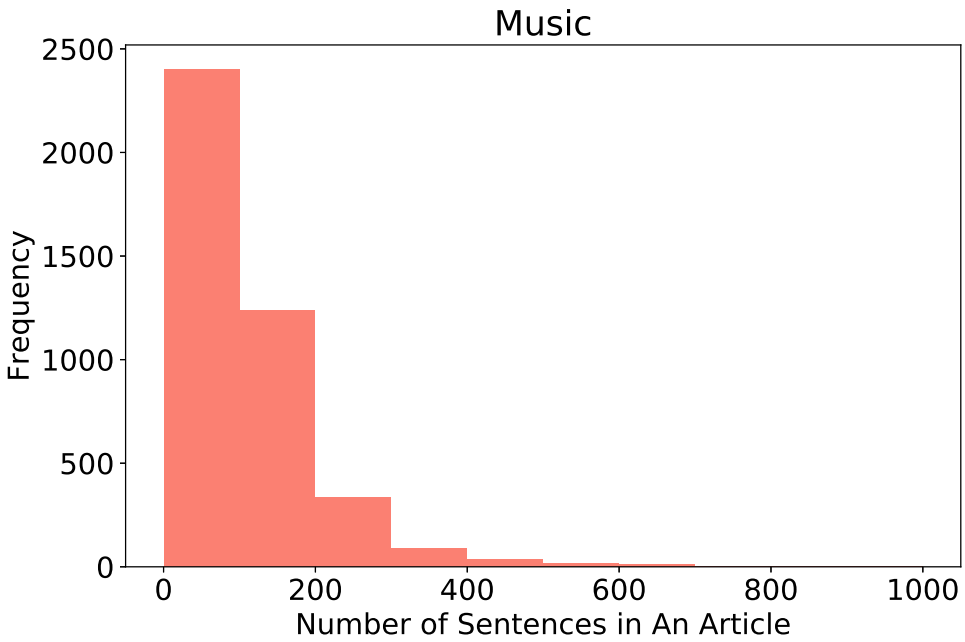


Figure A-25: This is a histogram of the number of sentences per article across all 4,125 articles in the Music category.

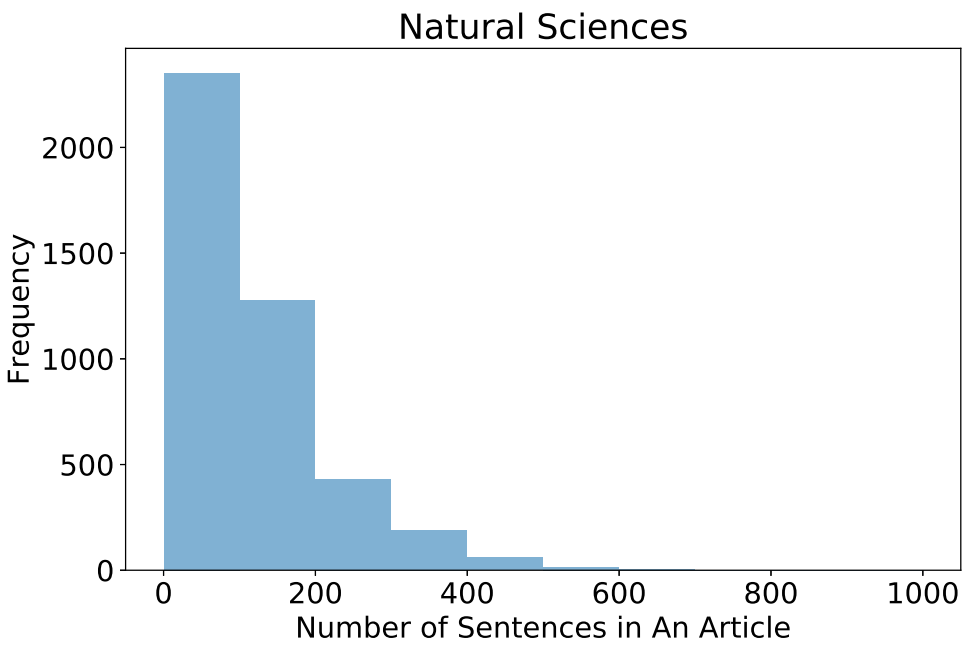


Figure A-26: This is a histogram of the number of sentences per article across all 4,329 articles in the Natural Sciences category.

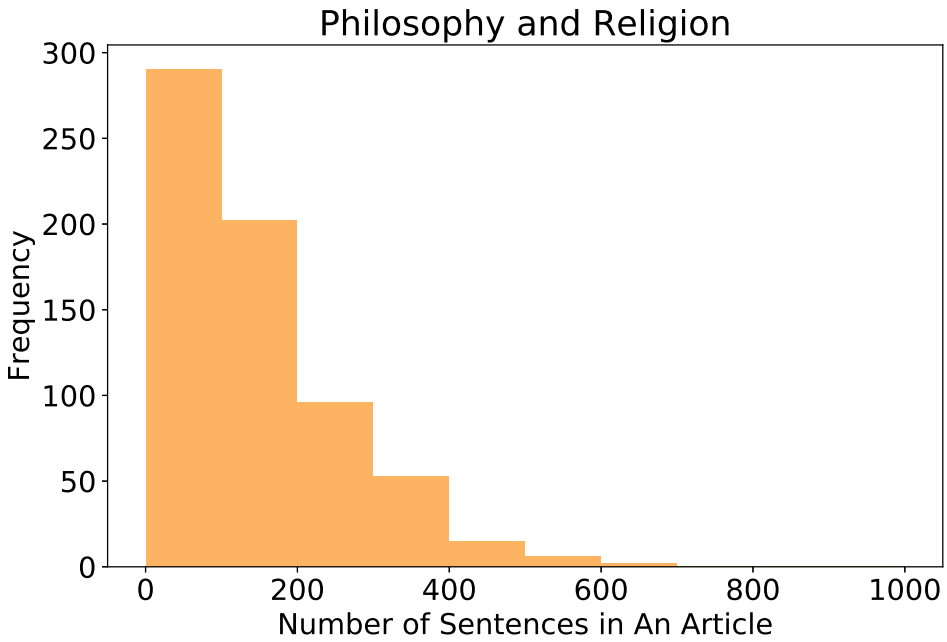


Figure A-27: This is a histogram of the number of sentences per article across all 664 articles in the Philosophy and Religion category.

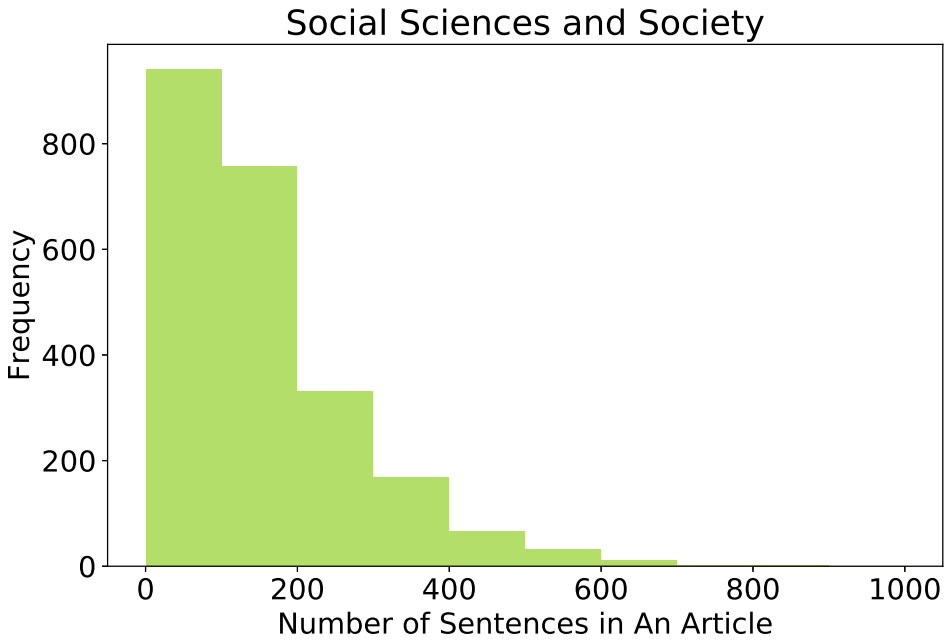


Figure A-28: This is a histogram of the number of sentences per article across all 2,313 articles in the Social Sciences and Society category.

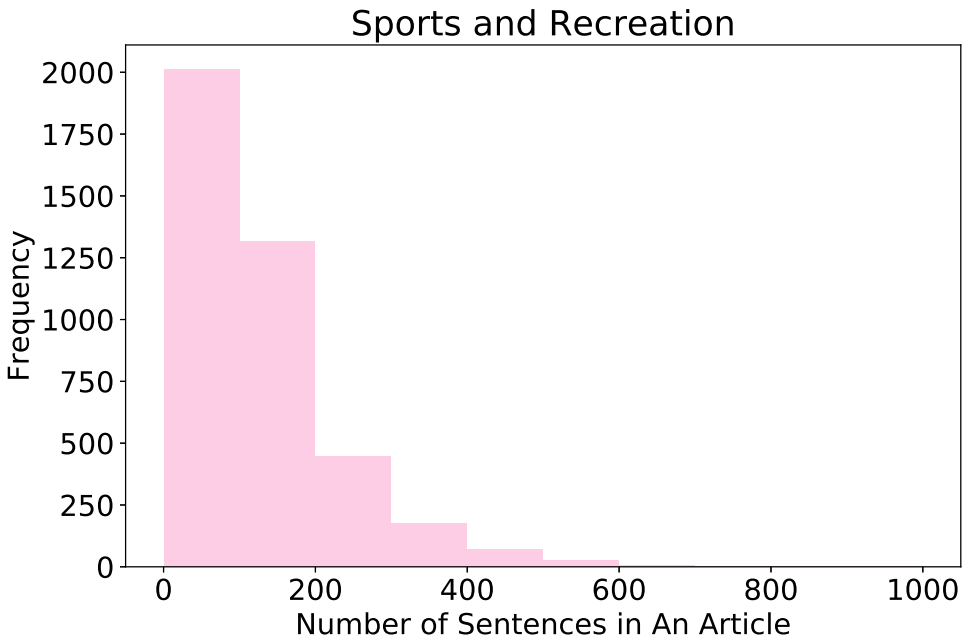


Figure A-29: This is a histogram of the number of sentences per article across all 4,059 articles in the Sports and Recreation category.

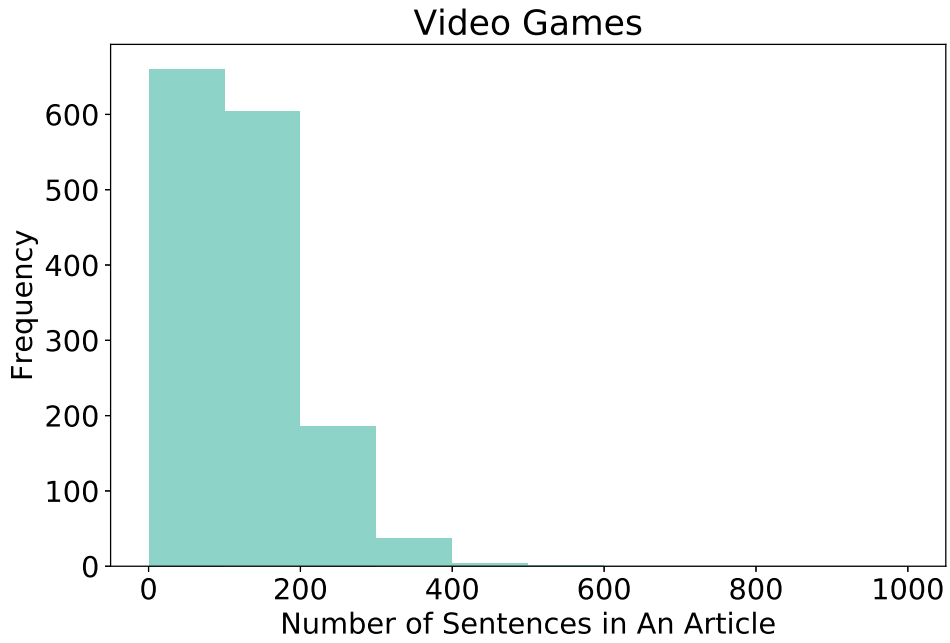


Figure A-30: This is a histogram of the number of sentences per article across all 1,492 articles in the Video Games category.

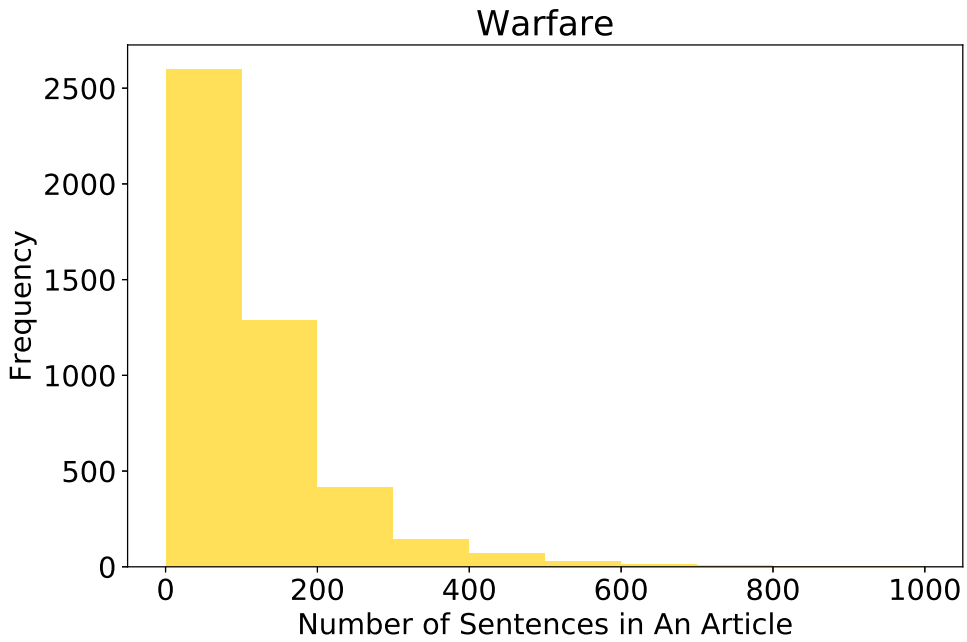


Figure A-31: This is a histogram of the number of sentences per article across all 4,556 articles in the Warfare category.

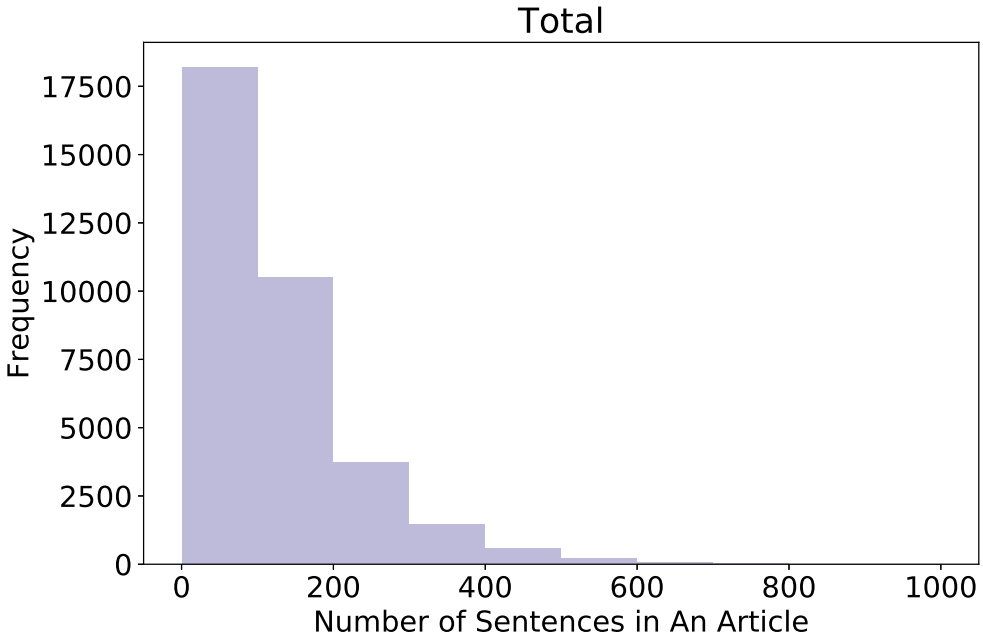


Figure A-32: This is a histogram of the number of sentences per article across all 34,786 articles.

Bibliography

H. Allcott and M. Gentzkow. Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, 31(2):211236, 2017. doi: 10.3386/w23089.

P. Atanasova, L. Màrquez, A. Barrón-Cedeño, T. Elsayed, R. Suwaileh, W. Zaghouani, S. Kyuchukov, G. Da San Martino, and P. Nakov. Overview of the CLEF-2018 CheckThat! lab on automatic identification and verification of political claims, task 1: Check-worthiness. In L. Cappellato, N. Ferro, J.-Y. Nie, and L. Soulier, editors, *CLEF 2018 Working Notes. Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum*, CEUR Workshop Proceedings, Avignon, France, September 2018. CEUR-WS.org.

R. Baly, M. Mohtarami, J. R. Glass, L. M. i Villodre, A. Moschitti, and P. Nakov. Integrating stance detection and fact checking in a unified corpus. In *NAACL-HLT*, 2018.

R. Bar-Haim, I. Bhattacharya, F. Dinuzzo, A. Saha, and N. Slonim. Stance classification of context-dependent claims. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 251–261. Association for Computational Linguistics, 2017.

J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

S. Dungs, A. Aker, N. Fuhr, and K. Bontcheva. Can rumour stance alone predict veracity? In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3360–3370. Association for Computational Linguistics, 2018.

P. Gencheva, P. Nakov, L. Màrquez, A. Barrón-Cedeño, and I. Koychev. A context-aware approach for detecting worth-checking claims in political debates. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 267–276. INCOMA Ltd., 2017. doi: 10.26615/978-954-452-049-6_037. URL `https://doi.org/10.26615/978-954-452-049-6_037`.

A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural networks : the official journal*

of the International Neural Network Society, 18:602–10, 07 2005. doi: 10.1016/j.neunet.2005.06.042.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8): 1735–1780, Nov. 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `http://dx.doi.org/10.1162/neco.1997.9.8.1735`.

D. Inkpen, X. Zhu, and P. Sobhani. A dataset for multi-target stance detection. In M. Lapata, P. Blunsom, and A. Koller, editors, *EACL (2)*, pages 551–557. Association for Computational Linguistics, 2017.

I. Jaradat, P. Gencheva, A. Barrón-Cedeño, L. Màrquez, and P. Nakov. Claimrank: Detecting check-worthy claims in arabic and english. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 26–30. Association for Computational Linguistics, 2018. URL `http://aclweb.org/anthology/N18-5006`.

E. Kochkina, M. Liakata, and A. Zubiaga. All-in-one: Multi-task learning for rumour verification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3402–3413. Association for Computational Linguistics, 2018.

T. Mihaylova, P. Nakov, L. Mrquez, A. Barrn-Cedeo, M. Mohtarami, G. Karadzhov, and J. Glass. Fact checking in community forums. AAAI Conference on Artificial Intelligence, 2018. URL `https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16780`.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

M. Mohtarami, R. Baly, J. R. Glass, P. Nakov, L. Màrquez, and A. Moschitti. Automatic stance detection using end-to-end memory networks. In *NAACL-HLT*, 2018.

P. Nakov, A. Barrón-Cedeño, T. Elsayed, R. Suwaileh, L. Màrquez, W. Zaghouani, P. Atanasova, S. Kyuchukov, and G. Da San Martino. Overview of the CLEF-2018 CheckThat! lab on automatic identification and verification of political claims. In L. Cappellato, N. Ferro, J.-Y. Nie, and L. Soulier, editors, *Proceedings of the Ninth International Conference of the CLEF Association: Experimental IR Meets Multilinguality, Multimodality, and Interaction*, Lecture Notes in Computer Science, Avignon, France, September 2018. Springer.

J. Pennington, R. Socher, and C. D. Manning. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL `http://www.aclweb.org/anthology/D14-1162`.

M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018. doi: 10.18653/v1/n18-1202.

D. Pomerleau and D. Rao. Fake news challenge, 2017.

J. Thorne and A. Vlachos. Automated fact checking: Task formulations, methods and future directions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3346–3359, Santa Fe, New Mexico, USA, Aug. 2018. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/C18-1283`.

J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal. FEVER: a large-scale dataset for fact extraction and verification. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018. doi: 10.18653/v1/n18-1074.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.

A. Vlachos and S. Riedel. Fact checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 18–22, Baltimore, MD, USA, 2014.