Otto-von-Guericke University Magdeburg

Faculty of Computer Science

Master Thesis

FACADE: Fake Articles Classification And Decision Explanation

Author:

Saijal Shahania

Supervisor:

Erasmo Purificato

26th July 2022

Reviewers:

Prof. Dr.-Ing. Ernesto William De Luca
AG Digital Transformation & Digital Humanities
Otto-von-Guericke-Universität Magdeburg

Prof. Dr. rer. nat. habil. Myra Spiliopoulou
AG KMD: Knowledge Management and Knowledge Discovery
Otto-von-Guericke-Universität Magdeburg

Shahania, Saijal:
FACADE: Fake Articles Classification And Decision Explanation
Master Thesis, Otto-von-Guericke University Magdeburg, 2022.

Abstract

In this thesis, we are designing an explainable system for detecting fake news. We are answering the question of which features distinguish fake news from real news and how we can use these features by differentiating them between low-level and high-level descriptors. This distinction is helpful in two ways: (a) it eliminates the need to calculate all of the features for all news articles, saving time and processing power; and (b) it provides insight into the fact that some fake news can already be detected using basic statistical measures. Our main contribution is, therefore, a cascading pipeline, where those two levels of features are employed consecutively. We also evaluate how we can choose features that keep the system ante-hoc explainable and how we provide this information to a user via visualizations. Attributing sentences to similar sentences from fake or real documents is our strongest high-level descriptor, allowing our system and the user to determine the truth value of parts of the news. It provides a potent classification accuracy and is a helpful tool for explaining why an article is fake or real. We can beat the baseline of TF-IDF by 4% and are just slightly weaker than a state-of-the-art method called *Roberta*. Furthermore, we created a UI that allows a user to get the classification and see the sentence attributions along with the most important features used for classifying individual articles. Therefore, our system provides an advantage over state-of-the-art methods like *Roberta* providing transparency while maintaining a high accuracy level. Henceforth, our system can be used in a scenario where a fact-checker or content manager has to decide the validity of an article quickly and effectively in their daily work. Overall, we can show that these explanations often give a good insight into the classification process and the factors making up fake news and can be extended further in the future using the insights gained.

Acknowledgments

Although I signed that I wrote this thesis independently, numerous people backed me up in this *Independence*. I would like to start by thanking my mom, without whose inquisitiveness, this thesis would have never happened. This all started with her asking me to explain why she should not share some fake WhatsApp forwards and how I know they are fake. Well. Rest is history! I would want to acknowledge my family for standing by all my decisions and helping me in every aspect they could.

Moving on, I would want to appreciate my supervisor Erasmo Purificato. He listened patiently to all the crazy ideas I threw at him and encouraged me to bring them to life, especially my obsession with Harry Potter. When it comes to the supervisors I had, I would also like to thank Vishnu Unnikrishnan and Sabine Wehnert. Vishnu always had a solution to my last-minute "I need help, I am stuck" voice notes and ensured I stopped being over dramatic. Sabine is responsible for me to dig deeper into the world of NLP, and I have such a cool experience working with her, always trying something new.

I would also like to thank Prof. Dr.-Ing. Ernesto William De Luca for allowing me to pursue my thesis under his chair and for his valuable inputs during the kick-off meeting. I want to thank Prof. Dr. rer. nat. Myra Spiliopoulou for being my mentor throughout my degree here at OVGU. I would be forever grateful for the learnings from her, especially about questioning every technique we present. She has a significant role in making me choose Research over corporate for my journey.

Nothing is bearable without a group of troubles that I call friends. I want to thank Arvind, Pawan, Vikram, and Sankul, who made me miss home a little less. Thank you for listening to my tantrums, providing me with *Ghar ka Khana*, and making sure that I stop overanalyzing every little thing. I would surely want to mention the oldest friends of my life Aditya and Ritwik who, for 11 years now, has been bearing my drama, and let us just say their favorite show now is *Keeping up with Shahania's* .

And well, this thesis would definitely be nowhere near what it is without the constant support for Marcus Thiel, who was a great teacher and the greatest friend I could have asked for. From listening to my 24*7 rants (read: slangs) to fulfilling my Indian food cravings around the clock, given the fact that he is actually a very german *German*. Thank you for understanding the not-so-jolly me and trying to comprehend my Punjabi. I think I should stop now to stop bloating my thesis any further, but how come I end without the potter quote, "Things we lose have a way of coming back to us in the end, if not always in the way we expect."

Contents

List of Figures

List of Tables

1. Introduction

The term *Fake News* has gained enormous popularity in recent years due to several lifestyle changes, the most significant of which is the invention of social media and its subsequent rise in popularity. The term gained even more traction following the historic US elections of 2016, in which Donald Trump might have had an advantage in the election results using misinformation and propaganda as depicted by Allcott and Gentzkow [2017]. Following this period, the term *Fake News* was selected as the year's word in 2017 by the American Dialect Society and Collin Dictionary, even though both organizations have different definitions of fake news. The American Dialect Society defines *Fake News* as "disinformation or falsehoods presented as real news or actual news that is claimed to be untrue"[1]. Collin has a more general definition: "false, often sensational, information disseminated under the guise of news reporting"[2].

In a recent scenario, fake news is also a propaganda tool in the Russia-Ukraine crisis. Both sides use it to trigger the audience, spread their worldview, and gain sympathy (cf. Alyukov [2022]). For example, in a recent incident wherein a famous American comedian, Joe Rogan, with over 9 million Twitter followers, posted that the American actor Steven Seagal is fighting against Ukraine as part of Russian special forces (cf. Figure 1.1). This story has been proven fake by showing that the image was just manipulated, but still, due to the number of followers Joe has, it might have reached wider audiences. When this is combined with the fact that the image shows the story being posted by a reputable news source CNN (as stated by Forbes[3]), it creates a piece of very credible news that may have spread further. In this case, however, it is merely amusing due to the quick identification and, in the worst-case scenario, would have had severe consequences for Steven Seagal. Other fake news in the war scenario may have an enormous impact and be more difficult to refute, resulting in more severe consequences such as demoralizing troops due to

[1] https://www.americandialect.org/fake-news-is-2017-american-dialect-society-word-of-the-year

[2] https://blog.collinsdictionary.com/language-lovers/collins-2017-word-of-the-year-shortlist/

[3] https://www.forbes.com/sites/berlinschoolofcreativeleadership/2017/02/01/10-journalism-brands-where-you-will-find-real-facts-rather-than-alternative-facts/?sh=3c3b4ecbe9b5

Figure 1.1: The figure shows the image posted by an American comedian Joe Rogan wherein a famous American actor Steven Seagal is claimed to be fighting as part of Russian special forces. This post gained tremendous popularity amongst trollers and people concerned about spreading fake news. However, later it was proved that the image had been manipulated. Taken from https://sports.yahoo.com/joe-rogan-mocked-sharing-fake-110905854.html.

news about weapons and troop sizes or gaining support through the incorrect spread of apparent cries for help.

Furthermore, as seen in the work of Ferrara et al. [2016], the emergence of bots in social media acted as fuel for the spread of fake news. They have been used extensively, for example, in the Gulf crisis (cf. Jones [2019]) as a spreader of misinformation and may act like humans as so-called "social bots" (cf. Shao et al. [2017]). Bots pose an important challenge to the problem of fake news since they can not only spread fake news much faster (and potentially more efficiently) than humans can, but they also exist in masses on social media. An extensive data study in 2019 by Liu [2019] showed that 16.9% of the Twitter accounts were deemed bots, which amounted to 31.2% of all tweets collected. Therefore third of all content was solely created by bots, which might only be the tip of the iceberg.

Therefore, timely intervention is necessary to avoid misinformation and bias. Unfortunately, although content managers manually check the news and other entries and fact-check websites/organizations are trying to discredit wrong information, many news/entries are missed. This is due to the sheer volume and speed of the emergence and spreading of fake news. For example, on Twitter, there are 500 million tweets tweeted daily[4], which technically have to be monitored for fake news and other unwanted content such as hate speech. Hence, standalone manual detection is not feasible, and content managers must be supported with automatic detection tools. For example, Meta currently employs an automatic methodology called *SimSearch-*

[4] https://www.internetlivestats.com/

Net++[5] that matches images against known fake news images, which then tags those posts as potentially containing misinformation. Fact-checkers then use this tag to see which posts need immediate attention.

1.1 Problem Statement

As discussed above, fact-checkers need a reliable tool to detect new potential fake news and mark the same for screening. Due to that, falling back on an existing known fake news corpus is not sufficient to keep up with new emergent fake news. Additionally, not all fake news is the same. Some news entries are blatant lies, while others hide their fake content among the facts. Therefore it is not enough to mark news as potentially fake but to disseminate its genuine and fake parts. In order to do that reliably, a huge corpus of facts would be needed for cross-checking. Since this is not realistic for the scope of this thesis and smaller content managers (such as bloggers), there is a need to find intrinsic or less corpus-heavy extrinsic factors that distinguish the fake parts of the news from the rest. Due to that, decisions have to be made transparently to increase trust in such an automatic system since such patterns must be cross-checked to be deemed false. For a non-specialist to do that, the criteria for finding those patterns need to be explained. Hence the goal of the thesis is to design an explainable automatic detection system that may mark news as fake, which may come from sources that: (according to the Penn State University Library[6], which is also used as defined by the creator of the Fake News Corpus[7]):

1. "[...] entirely fabricate information, disseminate deceptive content, or grossly distort actual news reports" or
2. "[...] come from a particular point of view and may rely on propaganda, decontextualized information, and opinions distorted as facts".

We call our system FACADE that stands for Fake Articles Classification And Decision Explanation.

1.2 Research Questions

For us to design our system FACADE, we have to decide on methods, i.e., models and algorithms, that help us detect fake news. The parameters of these methods need to be analyzed along with the properties of fake news to determine factors that influence the detection of such news. These factors might not be the same for all fake news, whereas we plan to sort the news into groups and check whether detecting fake news among those groups might be easier or harder for our system. Last but not least, the system has to be made available to an end user like a content manager of a social media platform. Therefore we need not just to mark the decisions made by our system as fake or not-fake, but also how we arrived at this conclusion using explainable features and post-hoc methods of explainability. Therefore our research questions are as follows:

[5] https://ai.facebook.com/blog/using-ai-to-detect-covid-19-misinformation-and-exploitative-content/

[6] https://guides.libraries.psu.edu/fakenews

[7] https://github.com/several27/FakeNewsCorpus

RQ-1 What methods are suitable for detecting fake news?

RQ-2 Which factors inside the corpus are relevant to fake news?

RQ-2a How do these factors influence the detection of fake news?

RQ-2b Are there groups that are easier or harder to detect using the aforementioned factors?

RQ-3 How can the detection process be made transparent and interpretable to a content manager or a fact-checker?

1.3 Structure of this Thesis

This section outlines how we organized the work in the subsequent chapters. To begin with, Chapter 2 gives an introduction to the fundamental techniques and algorithms we used to design our system FACADE.

Following that is Chapter 3, which gives a summary of the literary works on the term *fake news*, existing fake news detection systems, features used in the detection systems, and literary works on how to make the detection systems interpretable to the end users.

In Chapter 4 we give a brief overview of all the datasets used to perform the experimentations. Additionally, it shows results from the exploratory data analysis to give insights into the proposed datasets.

Moving on in Chapter 5 we discuss the methodology proposed to build our system. The chapter is organized to give a brief introduction of the concept design followed by a detailed description of different modules of the system, including the details about the handcrafted features for our detection system.

In Chapter 6 we show a glimpse of the user interface built by using our detection system with brief instructions on its usage.

Chapter 7 gives out the details about results post experimentation and compares the experiments with the baselines for evaluation purposes. Also, they cover the evaluation of standalone modules of our system.

The penultimate Chapter 8 analyzed the system's behavior on individual articles and checked how the system could be employed in the use case for the content managers.

Finally, in Chapter 9 we summarize the proposed work, followed by the insights derived, limitations, and future work.

2. Background

We used concepts from various techniques popular in the Machine Learning and Deep Learning communities to achieve the thesis' goals. We have grouped them into main categories, namely (i) Text Representation (ii) Clustering Algorithms (iii) Clustering Evaluation (iv) Text Segmentation (v) Classification Algorithms (vi) Deep Learning Trained Architectures (vii) WordNet

2.1 Text Representation

Our thesis works on distinguishing texts containing fake information from genuine one. In order to do this, we represent text in various forms to extract the distinguishing features. Some representations only capture the syntactical meanings, while others have the power to capture the semantics of the text. For our thesis, we work with specific popular representation methods such as TF-IDF, Doc2Vec, GLOVE, and BERT-based sentence embeddings. We discuss them briefly in the following subsections, along with the pre-processing steps of tokenization and lemmatization. Additionally, we present a methodology to find distances between documents (aside from general metrics like cosine similarity and Euclidean distance) using HOTT.

2.1.1 Tokenization and Lemmatization

Most textual features require the text to be broken up into *tokens*, i.e., smaller units, which are usually the individual sentences and words as well as punctuation, numbers, and similar. This process is called tokenization and has several challenges. Some - not all - of these challenges are:

- Detecting sentence boundaries, i.e., dots as separators alone, are not perfect since there might be abbreviations (e.g., Dr., Mr.), numbers (e.g., 8.5), or similar structures, which include a dot. Not all sentences end with a dot either.
- Detecting word boundaries, i.e., words might be compound words and, depending on the use case, need to be separated or kept together.

For example, more on tokenization and its challenges can be found in the paper by Webster and Kit [1992].

Additionally, some features that use word statistics might work better depending on the context and scenario if the words in the text are lemmatized, i.e., arriving at some root form. Those *lemmata* would e.g. map words like *working*, *worked*, *works* all to the same form of *work*. This is especially helpful if topics should be detected in a text since a topic is more about the general content of a document rather than its detailed semantics. More on lemmatization can be read in the paper by Plisson et al. [2004].

2.1.2 Part of Speech (POS) Tagging

Additionally, to understand the structure and word usage in a text, POS tagging can map words to their grammatical unit. Such grammatical units are tags like nouns, verbs, adjectives, and so on. This can help to understand the syntactic structure of a text, detect grammatical mistakes, and extract essential entities from a text like names, dates, corporations, and more. POS tagging is a big part of text processing, and the quality is not just dependent on the tagger model but also on the type of dataset used to train that model. Therefore, a good tagger trained in fictional books might behave badly in scientific texts. Most of the time, though, general purpose corpora are being used, accepting that the tagging will not be perfect. In our case, a general purpose tagger should be sufficient since these are often trained on corpora like Reuters[8], which are news articles and therefore exactly our domain. For example, more on POS Tagging can be read in a book by Voutilainen [2003].

2.1.3 Syntactic Text Embeddings

The use of the concept of term frequency and inverse document frequency (TF-IDF) embedding of the document in question is a proven method for extracting syntactical features. As the name implies, the term frequency refers to the number of occurrences of a specific term in relation to the document. The inverse document frequency, on the other hand, determines the term's uniqueness across the entire corpus (group of documents). The TF-IDF combines both components and assigns importance to the term based on its occurrence across documents. As a result, it can serve as a solid foundation for evaluating our syntactical features (cf. Ramos et al. [2003]).

Global Vectors for Word Representation (GloVe) are alternate methods of word representation to Word2Vec (cf. Pennington et al. [2014]). The advantage of GloVe over Word2Vec is that it, apart from the local context, also captures the global contexts to get the word vectors. The local context means that information extracted for the word depends on only the neighboring words and would not capture some semantic information. For example, in the sentence *The ball was on the ground*, Word2Vec would not be able to capture the information like if *the* is a stop word or not(cf. Mikolov et al. [2013]). The GloVe then finds the global statistics with the help of the co-occurrence matrix.

Doc2Vec is a popular method that represents documents in a fixed-dimensional vector (cf. Lau and Baldwin [2016]). It is built on top of Word2Vec, including the

[8] https://trec.nist.gov/data/reuters/reuters.html

additional context of the paragraph using the paragraph id. It is a very simplified representation of documents as it only captures the high-level semantics but not the granular level similarity.

2.1.4 Semantic Text Embeddings

We also explored the methods known to capture the semantics of the text. For this purpose, we used two models: Microsoft's *deberta-v2-xlarge-mnli*[9] and *all-MiniLM-L6-v2*[10].

DeBERTa (Decoding-enhanced BERT with Disentangled Attention) (cf. He et al. [2020]) builds over the top of BERT (cf. Devlin et al. [2018]) and RoBERTa (cf. Liu et al. [2019]), using two effective techniques: the disentangled attention mechanism and the enhanced mask decoder (cf. Vaswani et al. [2017]). Using the attention mechanism, each token is represented by two vectors that encode the token's content and position. Furthermore, disentangled matrices of the content and relative positions are then used to calculate the attention weights between the tokens. Finally, the enhanced mask decoder replaces the layer of softmax activation functions (cf. Sharma et al. [2017]) used in the output layer to increase the efficiency of predicting the masked tokens for training, as RoBERTa did with data from English Wikipedia[11], Book Corpus (cf. Zhu et al. [2015]), Open Web Text (cf. Gokaslan and Cohen [2019]), and Stories Corpus (cf. Trinh and Le [2018]).

On the other hand, *all-MiniLM-L6-v2* is a known sentence transformer model (cf. Reimers and Gurevych [2019]) used to capture the semantics of the text. The model uses the pre-trained *miniLM-L6-H384-uncased*[12] model and fine-tunes the same on 1B sentence pairs (cf. Wang et al. [2020]). The objective of the training is contrastive learning (cf. Khosla et al. [2020]), wherein if one randomly samples a sentence from an existing sentence pair, the model should be able to predict the sentence the sampled sentences were paired with. The two models are known to work best for capturing the semantic meanings of the text and, hence, are used in our thesis to get the sentence/segment/document embeddings. The DeBERTa takes relatively longer to run but is more accurate, whereas all-MiniLM results in faster computations. Based on the requirements in the thesis, they are used for different features.

2.1.5 Hierarchical Optimal Transport for Document Representation

IBM recently developed an exciting method called Hierarchical Optimal Transport for Document Representation (HOTT) to find semantic similarities between documents in a computationally efficient and interpretable manner (cf. Yurochkin et al. [2019]). The documents, according to HOTT, are viewed as a distribution of the topics, which are then viewed as a distribution of the words. The HOTT distance metrics employ the word mover distance (WMD), which allows us to capture similarities between documents even if they do not share common words (cf. Kusner et al.

[9] https://huggingface.co/microsoft/deberta-v2-xlarge-mnli
[10] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
[11] https://dumps.wikimedia.org/enwiki/
[12] https://huggingface.co/nreimers/MiniLM-L6-H384-uncased

[2015]). This works on the premise that the distance between two documents is the minimum cumulative distance that the words in one document must travel to match the distribution of the words in the second document.

The WMD is also seen as a special case for the Wasserstein distance (cf. Villani [2009]), the formula for which is shown in Eq. 2.1. Herein, C is the cost matrix where $C_{i,j} = d(x_i, y_i)$, d being the distance. The additional constraints make Γ an explanation for the transportation route between p and q. Using the Wasserstein distance, we can define WMD between documents d^1, d^2 as $W_1(d^1, d^2)$. Building on top of the same, the HOTT between the documents can be defined as seen in Eq. 2.2. Here $T = t_1, t_2, ..., t_{|T|}$ are the topics, and $\delta_{t,k}$ is the probability distribution for topic t_k. Then here, we use WMD over the topics and figure out how far the topics from one document have to travel to reach the topic distribution of other documents. This distance can be used to find the similarities and differences between the documents on the basis of latent documents and aid in explainability.

$$W_1(p, q) = \begin{cases} min_{r \in \mathbb{R}_+^{n \times m}} \sum_{i,j} C_{i,j} \Gamma_{i,j} \\ \text{s.t.} \sum_j \Gamma_{i,j} = p_i \quad \text{and} \quad \sum_i \Gamma_{i,j} = q_j, \end{cases} \tag{2.1}$$

$$HOTT(d^1, d^2) = W_1 \left(\sum_{k=1}^{|T|} \bar{d}_k^1 \delta_{tk}, \sum_{k=1}^{|T|} \bar{d}_k^2 \delta_{tk} \right) \tag{2.2}$$

2.2 Clustering Algorithms

For our analysis, we experimented with various clustering methodologies like K-Means, DBSCAN, Mean Shift, Affinity Propagation, Spectral Clustering, HDBSCAN, and Community Based Clustering. We will talk briefly about them in the following sections:

2.2.1 K-Means

K-means is the most widely used clustering algorithm in the machine learning community. It is based on the basic principle that elements within a cluster should be close to the cluster prototype (centroid), and elements from different clusters should be as maximally apart as possible (cf. Hartigan and Wong [1979]). Initially, we initialize randomly with k prototypes (centroids) wherein k is the desired number of clusters in the data. We then group the data points into their closest prototypes, wherein closeness is defined using the euclidean distance between two points. After all the points are assigned to a specific prototype, we recompute prototypes by averaging all the points assigned to the given prototype. This process is repeated until the prototypes do not change anymore. The main challenge for the K-Means algorithm is to find the initial centroids and the best value of k. The K-Means algorithm produces spherical, evenly sized clusters and hence is unsuitable for detecting all kinds of patterns. The optimization criteria for k means work towards minimizing the sum of the squared distance between the centroid and points within the cluster. In case we have instances $x_1, x_2, ..., x_n$ wherein each instance is a real vector having d dimensions, the goal of K-Means is then to partition the given n instances into k clusters (C) wherein $k \leq n$ such that we have a minimum sum of squared distance (J)

between the centroid (μ) and the points of the clusters, which is defined as seen in Eq. 2.3. The goal is then to find argmin (J).

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2 \tag{2.3}$$

2.2.2 DBSCAN

We saw that for K-Means, it is a challenge to find the initial centroids and decide on the best k. Furthermore, K-Means cannot deal with finding out the patterns have shapes other than spherical. For this purpose, we used another density-based spatial clustering algorithm for applications with noise (DBSCAN). The algorithm is built based on the neighborhood of the point of consideration. The number of points within one's neighborhood defines the density of a point, and DBSCAN works on the concept of density to form clusters (cf. Ester et al. [1996]). Let us take into consideration point p. The neighborhood of p is defined as a hypersphere having a radius ϵ with a center at p. The density of p is the number of points in its neighborhood, including itself. A *neighborhood* is defined as a dense one if the density of $p \geq$ minimum points (m) wherein m is a threshold. Based on this concept, we label all the points as core, border, and noise points.

- Core Points: The points with a density of at least m points are labeled as a core points.
- Border Points: The points which are not the core points but have a core point in their neighborhood are labeled as border points.
- Noise Points: All the leftover points are labeled as noise points.

Once all the points are labeled, the algorithm eliminates all the noise points. The next step is to connect all the core points within the distance ϵ of each other. This results in multiple groupings of connected core points, which are then different clusters. The final step is assigning the border points to the clusters with their respective core points. DBSCAN is known to work best with data samples that have patterns with similar densities. It does not favor a particular shape and is a robust algorithm for detecting non-spherical or arbitrarily shaped patterns. Although we do not have to define the initial centroids or number of clusters for the algorithm, choosing the accurate ϵ and m is also a big challenge since predetermining what distance makes an area dense needs certain pre-knowledge about the data.

2.2.3 HDBSCAN

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) is the extension of the DBSCAN algorithm by transforming it into the hierarchical clustering algorithm (cf. McInnes et al. [2017]). We saw in DBSCAN that ϵ and m are tricky parameters to estimate. For example, if the threshold for m is too large, many points would be labeled as noise points, and if it is too small, it might lead to everything being in a single cluster. On the other hand, it can be a

significant problem to have a single threshold for m in case we have clusters with varying densities hence HDBSCAN works on building hierarchies to figure out the order of merging dense areas. Once the merging is done, HDBSCAN decides to keep the big cluster or prefer its subclusters and can better determine the clusters that can capture the actual patterns.

2.2.4 Mean Shift

The Mean Shift Algorithm, just like K-Means, is a prototype (centroid) based algorithm, but unlike K-Means, it does not favor specific clusters nor requires us to set the initial number of clusters (cf. Comaniciu and Meer [2002]). As the name suggests, the algorithm works towards shifting the mean iteratively. The only parameter we have to define here is *bandwidth*, which is the radius of the local region where we search for the prototypes. Each point searches for its cluster in the Mean Shift algorithm by moving towards its prototypes in the decided bandwidth. As a result, the final destination of each data point is its centroid, which is determined by taking the weighted mean of all points within its bandwidth. Reaching this final destination is an iterative process; hence, it is called shifting the means. For instance, we start with point p and shift the point towards its mean m_1 in a given local region. Now we see the local region of m_2 and shift the mean again and do it until we reach the mean m_i (final destination), where the number of points in the decided bandwidth does not increase anymore. We can call the intermediate mean point a local solution and the final mean point a global one. The points that have the same final destination are then clustered together. Now say we have a given candidate centroid c_i at iteration t. This centroid is shifted as in Eq. 2.4:

$$c_i^{t+1} = m(c_i^t) \tag{2.4}$$

Furthermore, let us consider the neighborhood of chosen sample as $N(c_i)$, i.e., the area within the bandwidth of the c_i and m. The mean shift vector for each centroid, m, has its direction toward the area where we have the maximum density, which can be seen in Eq. 2.5. Here, K(c) is the RBF kernel (cf. Scholkopf et al. [1997]).

$$m(c_i) = \frac{\sum_{c_j \in N(c_i)} K(c_j - c_i) c_j}{\sum_{c_j \in N(c_i)} K(c_j - c_i)} \tag{2.5}$$

If we increase the local region (bandwidth), the local solutions will likely coincide with the global solution. Hence, setting a very high bandwidth disrupts us from finding the local patterns within the dataset because everything moves towards a single global mean point of the complete dataset. On the other hand, if we have small bandwidth, we might find some non-existing patterns, so the choice of bandwidth becomes very important for the mean shift algorithm.

2.2.5 Affinity Propagation

Affinity Propagation, like Mean Shift, does not require us to decide on a number of clusters in the sample. Instead, it works on the basic principle of attractivness (similarity) of each point to all other points. Here, each data point (sender) sends messages

to other points in the sample (targets) containing information about its relative attractiveness to all the other points (cf. Dueck [2009]). In response to this message, the target replies with its availability to associate with the sender. This decision is taken by considering the attractiveness of all the messages the target receives. Now, there is a follow-up message by the sender to all the targets with the revised attractiveness value based on the availability messages it previously received from its targets. This messaging between the sender and target continues until a mutual decision is made, resulting in the sender-target relationship. This target is called the sender's exemplar, and all the senders with the same exemplar form a cluster.

In this flow of message travel, we can categorize them into two main categories: responsibility messages and availability messages. The responsibility $r(i,k)$ (Eq. 2.6) acts as a gathered proof that target k is an exemplar for sender i. On the other hand, availability $a(i,k)$ (Eq. 2.7) is proof that sender i should consider target k as its exemplar, taking into consideration the values of all the targets. Hence, to be chosen as an exemplar, a data point has to be similar to many other data points, which is defined via the similarity matrix $s(i,k)$ and approved by the other data points to act as their prototype.

$$r(i,k) \leftarrow s(i,k) - \max_{k':k' \neq k} [a(i,k') + s(i,k')] \tag{2.6}$$

$$a(i,k) \leftarrow \min \left[0, r(k,k) + \sum_{i':i' \notin \{i,k\}} r(i',k) \right] \tag{2.7}$$

For Affinity Propagation, two essential parameters are needed, namely the *preference* and the *damping factor*, wherein the former limits the number of exemplars used. In contrast, the latter keeps checking responsibility and availability messages to avoid numerical oscillations while updating.

2.2.6 Graph Based Clustering

The thesis explored two types of graph clustering mechanisms: spectral clustering and modularity-based Louvain community detection.

2.2.6.1 Spectral clustering

Spectral clustering uses the fundamentals of graph theory, including making the similarity graph and reducing the data from higher dimensions into lower dimensional space (cf. Von Luxburg [2007]). Graphs are widely used to represent data because they are easily understandable. Graphs have two primary components, namely nodes and edges. The nodes can be used to represent the data points, and the edges act as a connection between the data points. Furthermore, we have the flexibility to have directed/undirected/weighted/non-weighted edges depending upon the use case. Additionally, the adjacency matrix (A) can be used to represent the graph wherein the indices of rows and columns represent the nodes, and the values are w_i*1 or 0, where 1 indicates the presence and 0 indicates the absence of edges between the nodes. w_i is the weight of edges and is one for unweighted edges. The graph theory's

important concepts are degree matrix (D) and Laplacian matrix (L). The degree of the node is determined by the number of edges it is connected with. The degree matrix is a representation of the degrees of the nodes in the graph wherein the value $(i.i)$ at the matrix position having i rows and i columns represents the degree of the node (cf. West et al. [2001]). The Laplacian matrix is also an alternate way of representing the graph, which is calculated just by subtracting the degree matrix and adjacency matrix (D-A). This matrix is the one that is mainly used for spectral clustering, wherein the non-diagonal points are the negative of weighted edge values, and the diagonals represent the degree of the node.

The next step is to find the eigenvalues of the graph laplacian that, in turn, help in graph clustering. There are two essential concepts to consider here: the spectral gap and the Fiedler value. The spectral gap is nothing but the first non-zero eigenvalues and gives us the idea of how dense a graph is, whereas the second eigenvalue is called the Fiedler value and its vector as the Fiedler vector, where each value of this vector tells us about the position of a node w.r.t. the decision boundary. For forming the clusters, the eigenvectors of the largest k eigenvalues are considered, and the data is then projected into a k-dimensional space.

Spectral clustering can help to find arbitrary shaped patterns and does not require us to decide on the initial centroid positions. Nevertheless, we need to define the number of clusters k corresponding to the k most significant eigenvalues.

2.2.6.2 Louvian Community Detection

Other graph clustering methods use community detection algorithms such as Louvain community detection, which uses modularity optimization (cf. Blondel et al. [2008]). Modularity is a metric used in graph structures that calculates the gain we get once the graph network is divided into small modules(communities or clusters). The higher the modularity, the denser the connections between the nodes within a community (more cohesion). Also, higher modularity indicates a sparse connection between the nodes belonging to different communities (more separation). This algorithm follows two steps wherein first it assigns each node (data point) to its own community and then sees the gain in modularity (Eq. 2.8) when the node is moved to other communities (cf. Blondel et al. [2008] and Traag et al. [2019]). In the event that there is no further gain in moving the node, the community is fixed for the node.

$$\Delta Q = \frac{k_{i,in}}{2m} - \gamma \frac{\sum_{tot} . k_i}{2m^2} \tag{2.8}$$

In Eq. 2.8, i is a node, m is the graph size, $k_{i,in}$ takes into consideration the sum of weights of edges from node i to nodes in community C. k_i considers the sum of weights of edges incident to node i and $\sum_{tot}$ takes into consideration the sum of edges incident to all nodes in C and γ controls the resolution.

The second step of the algorithm creates a new network considering the communities formed via the first steps of the algorithms. Then, the weights of the edges of the newly formed nodes are computed by summing up the weights of the edges between nodes of the two communities in question. The process is then repeated to form the more prominent communities, each having better modularity than the previous one.

2.3 Clustering Evaluation

Clustering is an unsupervised algorithm, i.e., we do not have a ground truth to compare the resulting patterns. Hence, it is a challenge to evaluate the results. The basic crux of good clustering is that a good algorithm gives us clusters having high cohesion (less distance between the points in the same cluster) and high separation (more distance between the points in the different clusters). There are two broad categories, namely extrinsic and intrinsic evaluation methods, that are used to evaluate the final clusters. The extrinsic evaluation techniques make use of the external data made available (cf. Rand [1971]). Since, for our thesis, we do not have any external information available, we instead use intrinsic evaluation measures such as Silhouette Score, Calinski Harabasz, and Davies Bouldin score.

The Silhouette Coefficient (s) is a popular method of intrinsic evaluation as it considers both the cohesion and separation in the clustering methods. First, it is calculated for each data point, then for a group, it is calculated by taking the mean of silhouette coefficients for all the data points (cf. Rousseeuw [1987]). The formula for calculating the same can be seen in Eq. 2.9.

$$s = \frac{b - a}{max(b, a)} \tag{2.9}$$

The separation is captured by b, calculated by taking the mean distance of the point in question to all the points in its nearest cluster. a, on the other hand, captures cohesion and is calculated by calculating the average distance between the point and all the points in its cluster. The range of s is between -1 and 1, wherein -1 means wrong clustering, 1 means very dense clustering, and values near 0 indicate the presence of overlapping clusters. One shortcoming of the method is that it has better values for convex clusters than for arbitrarily shaped ones; hence it might not be the best choice for evaluating density-based clustering algorithms.

The Calinski Harabasz index c is the ratio between the separation dispersion and cohesion dispersion (cf. Eq. 2.10). So, herein, the dispersion is nothing but the sum of squared distances (cf. Caliński and JA [1974]).

$$c = \frac{tr(B_k)}{tr(W_k)} \times \frac{n_E - k}{k - 1} \tag{2.10}$$

Here, n_E is the size of the data E,k is the groups into which E is partitioned, $tr(B_k)$ is the inter group dispersion trace and $tr(W_k)$ is intra group dispersion trace (cf. Eq. 2.11 and 2.12).

$$W_k = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^T \tag{2.11}$$

$$B_k = \sum_{q=1}^{k} n_q (c_q - c_E)(c_q - c_E)^T \tag{2.12}$$

Here q is the cluster in consideration, C_q are all the points in this cluster, c_q and c_E are the centroids of cluster q and E, respectively, and n_q is the number of points in q. The score is unbounded, and higher values indicate better clustering.

The Davies-Bouldin Index aims to describe the average similarity between each cluster C_i for $i = 1, ..., k$ with its nearest cluster C_j (cf. Davies and Bouldin [1979]). The index aims to have a balance between cohesion and separation via the measure R_{ij} which is defined in Eq. 2.13.

$$R_{ij} = \frac{s_i + s_j}{d_{i,j}} \tag{2.13}$$

Here s_i takes care of the cohesion by calculating the distance between each point of cluster i and its centroid. d_{ij} takes care of the separation by finding the distance between centroids i and j. The Davies-Bouldin Index (DB) is then defined as seen in the Eq. 2.14. The lowest possible value of the DB index is 0, and the values near 0 indicate better groupings of the data points.

$$DB = \frac{1}{k} \sum_{i=1}^{k} max_{i \neq j} R_{ij} \tag{2.14}$$

2.4 Text Segmentation

In the previous Section 2.1, we only discussed extraction of representations from documents as a whole. In order to analyze the texts at a different level of granularity, we also explored different text segmentation methods. This might help to get more detailed levels of text representations in order to understand the underlying structure and semantics of the news article. Segmentation in our thesis was done at three levels: sentence level, chunk level, and topic level. Specific approaches are available for breaking the documents into semantically meaningful segments (cf. Pak and Teh [2018]). We make use of the unsupervised setting and make use of lexical cohesion, and topic modeling.

Lexical cohesion determined the semantic relatedness of the group of words based on lexical frequency and distribution (cf. Morris and Hirst [1991]). The segmentation methods such as Text Tiling (cf. Hearst [1997]) are based on the principle that the shift in the subtopic is correlated to the lower lexical cohesion between the groups of words. This area of lower cohesion can indicate the topic boundaries. For our thesis, we use the technique called Topic Tiling (cf. Riedl and Biemann [2012]), which uses the core principle of Text Tiling. The only difference is that instead of analyzing the group of words, we analyze the topics and use Latent Dirichlet Allocation (LDA) as a topic model (cf. Blei et al. [2003]).

2.5 Classification Algorithms

In order to make use of the various features extracted from the text to decide if the text is fake or not, we experimented with various Machine Learning algorithms. We made use of both the standalone classifier algorithms and ensembles, such

as *Random Forest* (cf. Breiman [2001]), *Gradient Boosting* (cf. Schapire [2003]), *Decision Trees*, *XGBoost* (cf. Chen and Guestrin [2016]), *k-NN* (cf. Fix and Hodges [1989]), *SVM* (cf. Cortes and Vapnik [2004]), *Gaussian Naive Bayes* (cf. Raschka [2014]) and *AdaBoost* (cf. Schapire [2013]). The ensemble algorithms are just the combination of weak classifiers that together predict a new test instance (cf. Dietterich [2000]). They are known to have better generalization performance. This also showed up in our thesis with three main algorithms, namely *Random Forest*, *Gradient Boosting Ensemble* and *XGBoost* Methods performed the best in terms of accuracy as an evaluation measure, as seen in Section 7. We will discuss them in detail in the following subsection. Furthermore, we use feature importance scores to get deeper insights into what features have the maximum weightage towards classifier prediction (cf. Saeys et al. [2008]).

2.5.1 Algorithm Description

Random forest is a popular ensemble algorithm in the community (cf. Breiman [2001]). It is an ensemble of multiple decorrelated Decision Trees (cf. Rokach and Maimon [2005]). When a new test instance comes in, each base decision tree has a say via majority voting for the prediction. Say we have a dataset D with n instances and p attributes; the random forest can create different base trees by working around two main concepts of randomization: bootstrap sampling (cf. McCarthy and Snowden [1985]) and random attribute selection. The bootstrap ensures that we randomly sample n instances from D with replacement, and the random attribute selection ensures that for each base tree, only the random subset of attributes are selected as candidates, and the number is usually $\sqrt{p}$.

The *Gradient Boosting* algorithm is based on the principle of boosting (cf. Schapire [2003]), wherein the first model is built just on the training data, and then the second model is trained to correct the incorrect predictions made by the first model (cf. Natekin and Knoll [2013]). It creates multiple models like this in a sequence where the subsequent models work towards improving the performance of their predecessors. This improvement is made by building the subsequent models on the errors/residuals of their predecessors. The end game is then to optimize the loss function, which is log-likelihoods in the case of the classifier setting. The optimization is achieved by adding the weak base classifiers using gradient descent (cf. Ruder [2016]).

XGBoost is the extreme gradient boosting algorithm which follows the same principle of boosting as in *Gradient Boosting* (cf. Chen and Guestrin [2016]). The difference comes in terms of regularisation strategies wherein *XGBoost* additionally use $L1$ and $L2$ regularisation techniques that help improve the generalization performance of the classifier. Moreover, training in XGBoost is faster than *Gradient Boosting* because it can be parallelized.

2.5.2 Feature Importance

Once we have selected the best classifier that helps find the best decision boundary to distinguish between the fake and real texts, we can also find the features that maximally contributed to the classifier's decision. Ranking the features according to their contribution can be helpful in various ways. First, it aids in the understanding

of the data better, hence contributing implicitly towards explainability. It helps us better understand the link between the features and the predicted class and helps discard the irrelevant features, resulting in faster computations. Hence, it reduces the size and time complexity of the classifier. The method used in our thesis performs the permutation-based Feature Importance (cf. Altmann et al. [2010]). It is based on observing the rise or fall in the errors when the features are permuted. The tremendous change in the error value on permuting the feature indicates the greater importance of the feature to the classifier. Permutation-based feature importance is used for our thesis because it is a straightforward model-agnostic approach.

2.6 Deep Learning Trained Architectures

In this thesis, we also made use of specific pre-trained Deep Learning Architectures for extracting semantic level features that, in turn, help distinguish between fake and real news. Furthermore, different models are trained on different datasets depending on the use case. The architectures specialize in different use cases as discussed below:

2.6.1 Natural Language Inference

To figure out if the sentences in the text are following each other (mostly in real news) or contradicting each other (mostly in fake news), we used the concept of Natural Language Inference (NLI), which, given two texts, finds the links between them (cf. MacCartney [2009]). In order to do the same, it makes use of the concept of *premise* (p) and *hypothesis* (h) to give out the labels of *entailment* (h is true), *contradiction* (h is false), and *neutral* (h and p have no relation). Our training model is RoBERTa's Multi-Genre NLI[13] (Robustly optimized BERT approach) (cf. Liu et al. [2019]). This model is trained in the footsteps of BERT's pretraining sequence, where there is heavy dependence on large amounts of text consisting of 5 English language corpora across different domains, namely the Book Corpus and English Wikipedia, CC-News[14], Open Web Text, and Stories. In addition, the GLUE (The General Language Understanding Evaluation) dataset is used, a collection of 9 different datasets built for evaluating natural language understanding systems to evaluate the work (cf. Wang et al. [2018]). The training of BERT has two primary goals: masked language modeling (MLM) (cf. Nozza et al. [2020]) and next sentence prediction (NSP) (cf. Shi and Demberg [2019]). The main objective of masking is to increase the generalization power for prediction by optimizing the number of predictions for the masked tokens. NSP, on the other hand, works on predicting if the two consequent texts follow each other or not, which, in turn, aids in improving the performance of Natural Language Inference by figuring out the links between the pair of sentences.

2.6.2 Sentiment Analysis

Specific, unique models are used to figure out the labels of sentences in terms of sentiments like positive, negative, and neutral. The model used for performing the

[13] https://huggingface.co/roberta-large-mnli
[14] http://web.archive.org/save/http://commoncrawl.org/2016/10/newsdataset-available.

task is *distilbert-base-uncased-finetuned-sst-2-english*[15]. The DistilBERT is also a transformer model, which is the distilled version of the BERT model and is small and fast compared to the basic BERT (cf. Sanh et al.). The training data is the same as that of BERT, namely the Book Corpus and English Wikipedia. However, it was done in a self-supervised (cf. Zhai et al. [2019]) manner in which pretraining was done on the texts without labels, and the labels were generated using the BERT base model, allowing them to use a large amount of the corpus. This model's objective was threefold: the distillation loss, MLM, and cosine embedding loss. The distillation loss aimed at returning the probabilities similar to the BERT base model, whereas the cosine embedding loss aimed to create the latent states same as that of the BERT (cf. Aguilar et al. [2020]).

2.6.3 Zero Shot Learning

Zero-Shot Learning (ZSL) is based on the concept of Transfer Learning (cf. Weiss et al. [2016]), which has been gaining popularity since the emergence of many unsupervised learning algorithms. The basic crux of the ZSL lies in learning (training) as a classifier using a specific set of labels and then evaluating the learned model using a completely different set of labels that the classifier has never seen before (cf. Romera-Paredes and Torr [2015]). This makes the classifier perform tasks it was not trained to do explicitly. For our thesis, we use the traditional ZSL, which requires us to initially provide a set of labels with which we want to categorize our data. We use Facebook's *bart-large-mnli*[16], which is used as a ZSL classifier. The model works on the principle of *premise* and *hypothesis*, wherein the label we want to attribute a text to is the hypothesis, and the text itself is the premise. For example, suppose we want to label the text for its topics and determine whether the topic being discussed is a sports topic or not. In that case, we create a hypothesis that the text discusses sports and then calculate the probabilities for entailment and contradiction of the hypothesis, which are then converted into labels.

The BART model (cf. Lewis et al. [2019]) is a transformer encoder-decoder with a bidirectional encoder similar to the BERT and an autoregressive decoder similar to the GPT (cf. Radford et al. [2018]). BART is pre-trained using the Book Corpus and the English Wikipedia. There are two significant steps involved in BART's pretraining, i.e., using a random noising function to deliberately corrupt the texts and then learning the classifier to reconstruct the original text.

2.7 WordNet

We use the lexical knowledge of WordNet[17] in our Thesis to capture the syntactical and semantical features of the text. The main reason for working with WordNets is that they can form relationships with the different forms of words and between particular word senses, which in turn aids in semantic disambiguation (cf. Navigli [2009]) between very similar words. Furthermore, we have a prevalent method called the Lesk algorithm (cf. Lesk [1986]) to deal with word-sense disambiguation (cf. Ide

[15] https://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english
[16] https://huggingface.co/facebook/bart-large-mnli
[17] https://wordnet.princeton.edu/

and Véronis [1998]). It is made on the concept that words in a given section of the text, also referred to as *neighborhood* of the word, primarily likely belong to the same topic. The Lesk algorithms have been made to work using the structures of WordNet (cf. Banerjee and Pedersen [2002]).

Another main advantage of WordNet over traditional lexical databases like Thesaurus is that semantic patterns between the words are explicitly labeled. The primary connections in the WordNet are based on exact meanings (synonyms). The synonyms are then grouped into disordered sets called *synsets*. There are various relations that can be derived from the synsets. The ones that are the main focus of the thesis are the super-subordinate ones like *hypernyms* and *hyponyms*. The hyponyms are the specific words of a given category; for example, *lawn tennis* is the hyponym of the word *sports*. The hypernym is the more general form of the word; for example, a *car* is a hypernym for *BMW*. The synsets then link the hypernyms to the hyponyms. The other essential groupings we use are parts of speech categories. We have synsets organized as nouns, verbs, and adjectives.

For nouns and verbs, several categories are defined based on their type. The noun and verb synsets are arranged in hierarchies wherein the root node for the noun hierarchy is called *Entity*. For the verb hierarchies, the bottom nodes are the troponyms that focus on moving towards specific manners, for example, *walk-march*. The adjectives, on the other hand, are grouped based on antonyms. For example, the pair *wet-dry* can be seen as showing high semantic contradictions. Furthermore, each element of the antonym pair is linked to the elements it is semantically closest to. For example, *dry* is linked to *shriveled*, and *wet* is linked to *drenched*. Additionally, we have the concept of satellite adjectives, which are the adjectives that are used to define a particular characteristic of the leading adjective group. For example, the word *color* is a central adjective, whereas the word *green* is a satellite adjective.

3. Related Work

3.1 Literary View on Fake News

The term *Fake News* is used in many different contexts based on its characteristics and has been categorized accordingly. As shown by the studies of Balmas [2014] and Day and Thompson [2012], before the popularity of *Fake News* and social media, the most common category of supposedly fake news was political satire. Back then, political satires were also seen to confuse the viewers, discourage voting and defame some leaders, as shown by Baumgartner and Morris [2008]. Today, satires are mostly seen as humorous content; instead, *clickbait* is emerging as a new menace. According to Chen et al. [2015], sensational headlines are one of the primary reasons for the spread of misinformation and should fall under *clickbait*. The fake part of such news is not necessarily the main content of the news but mostly the juicy headlines that are solely written to get users' attention and hence increase their views on the same. Our thesis focuses on *Fake News* which is deliberately deceptive or highly biased reporting. As seen in the work of Calvert and Vining [2017], fake news that deliberately introduces false claims and distorts the truth can have first-hand, significant, and prompt effects and should be addressed at the early stages of its emergence. On the other hand, extremely biased news, i.e., the news that is written keeping in mind a particular point of view or the ones where a person's opinions are stated as facts, is even worse. The primary purpose of such biased news is to provoke people and spread propaganda. For example, the work by Aggarwal et al. [2020] shows how biased news conditions people's thinking and makes them believe facts that are not true. The problem with biased news is that it is tough to fact-check; if not detected in the early stages, it can create a very opposing view of a specific topic.

The direct impacts of the fake news categories mentioned above can be seen in almost every field, such as economics, politics, health care, and entertainment. In the work of Allcott and Gentzkow [2017], it is seen that fake news tends to impact the user's buying behavior and hence help the business make profits. A prevalent example of the same is selling products that claim to reduce weight in one week. On the political front, as seen in the work of Coe et al. [2008] and Nelson and

Taneja [2018], fake news can significantly impact the democratic voting process by influencing public opinion towards a particular candidate. The 2016 US elections are the most prominent example of the influence of fake news in this field. The most influential impact can be seen in healthcare because most users lack knowledge about a specific domain and fear the consequences. The works of Seltzer et al. [2017] and van Der Linden et al. [2020] show the impact fake news had in times of Zika and coronavirus outbreaks, respectively. Another intriguing study by Zimet et al. [2013] shows the enormous impact of fake news on vaccinations. Bryant and Levi [2012], Chipeta et al. [2010], and Rowlands [2011] discuss the implications for contraception, abortion, and women's health. Duffy et al. [2020] in his work discusses how even interpersonal relationships are becoming the target of fake news. The work highlights the link between news sharing and a well-informed person's status in society. They talk about how the news originates from a few influential people termed *opinion leaders* and is spread by their *opinion followers*. The interpersonal element is damaged on both fronts because the influential people lose their credibility amongst the followers, and the followers lose this status of being well informed amongst the people in their echo chamber (cf. Cinelli et al. [2021]).

These works display a multitude of fields being impacted. Polarised opinions are an alarming threat to society. What makes it an even bigger problem are the channels through which the news spreads, i.e., not only are there people who knowingly or unknowingly become the originators of fake news but there are also people who increase the magnitude by acting as spreaders. For example, in work by Vosoughi et al. [2018], they studied how humans (not bots) contribute to the faster and greater spread of fake news. They attribute this to fake news's unusual and exciting nature, which makes people share it more. This novel characteristic of fake news is what makes it more alarming. Furthermore, as seen in the work of Adnsager et al. [2001], the very well-written fake news is the one that exists together with the actual news. This is done to trick the audience into figuring out whether the parts of the news are fake or real, which leads to a higher probability of its spread as it is less likely that everyone takes efforts to fact check at such granularity. Furthermore, this mix of fakeness and realism also makes the news persuasive and more likely to be acted upon. This characteristic of fake news is mainly used to spread propaganda or as the main crux of hate speeches. Hence, this kind of not-so-obvious fake news needs attention, and detecting it is the main focus of our thesis. This also motivates analyzing the articles not as a whole but at different levels of granularity to capture the parts of fake text.

Therefore we need to investigate how current research and existing systems deal with fake news and its detection, which leads us to the following section, discussing existing fact-checking/fake news detection systems.

3.2 Existing Detection System

With the rise in the spread of fake news, a multitude of research is being done to find the origin of the same, find ways to stop the spread and find the characteristics peculiar to fake news. The work started with manual fact-checking initiatives and is now moving towards automatic detection and fact-checking methodologies. An

exciting piece was proposed by Conroy et al. [2015] to compare the approaches above. The most famous manual fact-checking system in place is *Truth-o-Meter*[18] by Politifact, which focuses mainly on political news. They believe that everything is not wholly fake or real; hence, their ultimate goal is to figure out the relative truthfulness of a statement. So they divide the news into six groups, ranging from complete truth (*True*) to completely fake (*Pants on Fire*). Another quite famous work is by *Snopes*[19], which fact-checks not only the political stuff but also anything that is trending. They also work towards verifying different content sources like images and videos.

There also exists an exciting project by Duke Reporter's Lab called *fact-checking*[20], a database used to track non-partisan groups. The database aims to have a pool of well-informed resources in one place for the users, which makes it easier for the public to verify the facts before sharing them with others and, in turn, slow down the spread speed of fake news. On the automatic detection front, some works have built upon the existing works of deception detection. However, fake news detection is not precisely similar to deceptive detection regarding the motivation behind both, where the former seeks more financial or political gains. In contrast, the latter works more towards personal protection. The similarity is more in terms of content having similar characteristics in terms of linguistics, as seen in the work of Pérez-Rosas et al. [2017]. There is a comprehensive work done in deceptive verbal detection by Fitzpatrick et al. [2015] upon which the work by Pérez-Rosas et al. [2017] is built. In our thesis, we took motivation for specific linguistic approaches as features and considered them the first phase in our detection system, FACADE.

Vlachos and Riedel [2014] proposed that finding the news that needs attention, verifying the same, and then predicting if it is fake or not can all be solved using Natural Language Processing tasks. Hence, the detection system should be built in the form of sub-components wherein each component works towards solving smaller sub-tasks cumulatively, predicting whether the news is fake. We used these modalities in designing our system, FACADE.

The initial works of the detection system were built as a supervised learning problem, which was solved using features like *up-votes* as seen in Aker et al. [2017], entity and verb form as seen in Zuo et al. [2018] and only lexical and syntactic features as seen in Zhou et al. [2020]. As a result, these systems gave faster results and used fewer computation resources.

The state-of-the-art detection systems rely on the power of deep learning, which also tries to capture the context in which the content is published, as seen in the work of Zubiaga et al. [2016], which showed that rumors could be detected by analyzing how the information is debated and shared. In the latest work by Zhang et al. [2021], they have seen the relationship between the emotions portrayed in the news content and the end users' emotions expressed in the comments made on the content. Therefore, they have proposed using the *dual emotion features* in the existing fake news detection systems to enhance their performance. Another work by Kipf and

[18] https://www.politifact.com/truth-o-meter/
[19] https://www.snopes.com/fact-check/
[20] https://reporterslab.org/tag/fact-checking-database/

Welling [2017] wherein the detection system aims to study the propagation behavior of a rumor using Graph Neural Networks.

Summarising the works, we can say that the manual and automatic fact-checking/fake news detecting systems are both the need of the hour and should be used to complement each other. The rate at which news is published and spread makes it difficult to verify them manually at the appropriate time as it is very time-consuming (cf. Hassan et al. [2015] and Adair et al. [2017]). This makes an automatic detection and checking system critical since it can act as a pre-filter to reduce the amount of content to be analyzed by the fact-checkers. Detection systems can filter out very obvious fake and real news, and the not-so-certain cases can then be shown to checkers that enable faster verification. One cannot ignore fact-checkers importance and entirely rely on automatic systems because systems can be prone to errors, lack domain knowledge, and need additional feedback given the stakes. The need for the domain expert to validate the content has been highlighted in the work of Ha et al. [2021]. Our system works on this principle of accommodating the automatic detection systems with additional fact checkers (content managers). FACADE includes the extra element of explainability to act as a quickly understandable pre-filtering tool for content managers.

3.3 Linguistic Features

Different automatic detection systems rely on different characteristics of fake news to differentiate them from the real ones. With the changing times, the features looked upon have moved towards combining semantics with linguistics and syntax. In the early works like those by Rubin et al. [2016], the focus is on using the power of linguistic approaches that capture the content's writing style. The features that stood out were punctuation, parts of speech, and the types of words we utilized in the first phase of the detection system. Shu et al. [2017] show how the features can be derived from what we define as the article's content. They constitute source, headline, body, and other videos/images as the meta information that has to be analyzed to find the distinguishing characteristics. For our thesis, we mainly used the content from headlines and the body of news articles to extract the linguistic features. The sources are analyzed briefly as part of exploratory data analysis to understand what kinds of newsgroups tend to be more inclined toward posting fake news. The work also discusses categorizing linguistic features into lexical and syntactic features. The lexical features focus on the word-character statistics, such as average word length, type-token ratio, and article length. On the other hand, the syntactical features go a step further in analyzing the sentences as a whole by analyzing the parts of speech usage, punctuation, n-grams, and other methods that consider the articles as a bag of words as Term Frequency-Inverse Document Frequency (TF-IDF). In the first detection phase, we include both lexical and syntactical features for our detector.

The more recent works of the fake news detection techniques work on capturing the semantics and the discourse. Herein, semantics refers to the meaning of individual words or sentences and, therefore, the context in which they stand. In contrast, discourse refers to the overall argumentation structure of the text, which requires not just semantics but also knowledge about the relationships between sentences. As

seen in Li et al. [2014] and Pérez-Rosas and Mihalcea [2014], the semantic features in most works revolve around the 80 semantic classes defined by Linguistic Inquiry and Word Count (LIWC)[21] (cf. Pennebaker et al. [2001]). However, Pisarevskaya [2017] used Rhetorical Structure Theory (RST) (cf. Mann and Thompson [1988]) in a more advanced work to extract the features at the discourse level for distinction.

There are also works making use of attribute-based features that directly use the concept of the deception theory as seen in Siering et al. [2016], the main crux being the pronounced difference between the facts and fantasy (undeutsch hypothesis-Undeutsch [1967]). The survey by Zhou and Zafarani [2018] categorized the attribute-based features into ten different subparts, each focusing on a different aspect of textual content. The features as part of the categories Diversity (Lexical Diversity (Malvern et al. [2004]), content word diversity, and redundancy), Informality (Misspelled words), and Readability (Flesch-Kincaid (Kincaid et al. [1975])) have been explored as part of our thesis by taking motivation from the works of Zhou et al. [2004] and Afroz et al. [2012].

The more recent works focus on handcrafted or domain-specific linguistic features that can help unravel the characteristics of fake news. The work by Potthast et al. [2017] focuses on the stylometric aspect of the content to unravel the bias and fake news. Their work shows that it is not easy to deceive using the writing style, and certain aspects indicate the difference between a balanced and a biased news article. They show the same by analyzing the texts from the left-wing and right-wing supporters and seeing an uncanny resemblance of extremism in both texts. This work motivated us to look into the writing aspects since, for our thesis, we have also included biased news as part of the fake news detection. Another exciting work by Feng et al. [2012] uses probabilistic context-free grammar (PCFG) to detect writing styles (cf. Jelinek et al. [1992]). This motivated us to study the parts of speech rules for fake and accurate news and then use them to find the distinctions.

The recent works mainly focus on knowledge-based features, which also add more credibility to the systems and add the explainability aspect to the same. For example, the state-of-the-art dataset FEVER[22] is widely used as an external database for fact extraction and verification (cf. Thorne et al. [2018]). The very crux of the dataset, wherein each claim is labeled as either *Supported*, *Refuted*, or *NotEnoughInfo*, is used to design one of the essential features termed *Attribution* (cf. Section 5.4.1) for our detection system. The motivation for using external datasets for fact-checking comes from works by Magdy and Wanas [2010] and Hassan et al. [2015]. In another exciting work by Etzioni et al. [2008]; open web sources are used to validate the claims in terms of consistency. Finally, in the works by Shi and Weninger [2016], Wu et al. [2014] and Ciampaglia et al. [2015] used the knowledge graphs to make the relationship between the claims in the news articles and the already existing factual information.

[21] https://www.liwc.app/
[22] https://fever.ai/dataset/fever.html

3.4 Style and Knowledge based Features

Apart from the well-established feature categories for fake news detection, some particular concepts were used to explore fake and real news characteristics. One such concept in our thesis revolves around stance classification, which in theory tends to figure out, given the claim and the article, that the article supports, rejects, or is irrelevant to the claim. The reasoning has been captured as a text entailment problem in work by Dagan et al. [2010], wherein external sources are present to check the claim against. We use this entailment within the articles by checking if the entailment or contradiction within the articles has any relationship with the text being fake or real.

Another work by Jin et al. [2016] uses the stance features to find the opinions of the features and users on the news article as an entailment setting to determine whether the users support the article. Finally, moving on towards exploring the latest features that help distinguish the news articles, there is work by Ma et al. [2016]. They use topics derived via Latent Dirichlet allocation to detect rumors from microblogs. We have used this as a motivation to explore whether fake and real articles favor specific topics. Some works have extracted the features in an unsupervised setting, like in the work by Ruchansky et al. [2017] and Ma et al. [2015], who use the latent text embeddings of the articles. A similar approach is used to represent the content of the articles in our thesis.

There are also knowledge-based features that are gaining popularity that consider the opinions of expert fact-checkers in evaluating the credibility of the predictions. One such technique is crowdsourcing-oriented fact-checking as shown by Shu et al. [2017], wherein the intelligence of the users are used to teach the mass to annotate the news article's content. The two most famous initiatives using the concept of crowdsourcing are Fisskit[23] and LINE[24]. Fisskit is an initiative built on the principle of *fisking*; a famous method bloggers use to review articles sentence-wise. Herein, they provide the platform for the end users to annotate the articles per line. This paper was the groundwork for the second phase of our detection system, wherein we analyzed the text at different levels of granularity and, to do so, worked with various text segmentation techniques.

Furthermore, another work by Hosseinimotlagh and Papalexakis [2018] focuses on finding patterns within different types of fake news based on the content with the help of clustering methods. This was used in our thesis as the foundation of the pipeline's second phase to find the different groups of news articles within which further experiments were conducted to distinguish between real and fake articles. The motivation for exploring the graph-based methods was taken from the work of Gangireddy et al. [2020], wherein graph networks are used to figure out the differences between fake and real articles using bi-clique similarity.

3.5 Explainability

In the previous sections, we saw tremendous work in the field of fake news detection using features that made use of linguistics, semantics, domain knowledge, external

[23] https://fiskkit.com/
[24] https://grants.g0v.tw/projects/588fa7b382223f001e022944

factual knowledge, and readers' opinions. With the proliferation of methodologies, the need for reliable systems is becoming increasingly important. The user should be able to trust the system. In general, there is much work being done to make Machine Learning explainable, which aims to achieve three main goals, namely (i) Transparency (ii) Lucidity (iii) Possibility to question the system. The degree of explainability is perceived at a global and local level. The former aims to give an overall view of how the features are responsible for the model's prediction, and the latter provides the explainability at an instance level. Specific models such as decision trees are used as surrogate models to explain the predictions of black box models such as neural networks. The approaches to explainability are also subdivided into model agnostic and model-specific ones. The model-agnostic approaches are famous because they can be used irrespective of the model chosen (LIME, SHAP). The model-specific ones are, on the other hand, dependent on the model functions. The work by Molnar [2022] summarises the fundamentals of explainability in great detail.

As said in Ha et al. [2021] and Zhou et al. [2019] because of the coexistence of fake and real news, there is a need to incorporate the extra eye of the audience which can be achieved via explainability. The state of the artwork on this front is shown in work by Shu et al. [2019], wherein they have proposed a system called *dEFEND* that uses a deep hierarchical co-attention network to represent the textual features. Additionally, they use the network to figure out the relevant sentences in the articles and the follow-up comments of the audience that were mainly responsible for detecting it as fake news. We use this feature of sentence-level explainability to make our system more interpretable.

In another exciting work *Xfake* by Yang et al. [2019], attributes, semantics, and linguistics are used for the detection. The explainability aspect stems from self-explainable features aided by visualizations, which also serve as the foundation of our detector's explainability module. They visualize the features' importance in predicting fake news and the significant phrases and words. The work by Silva et al. [2021] focuses on explainability for early detection of fake news using hierarchical attention networks. The work by Reis et al. [2019] uses SHAP values for the aspect of explainability. Finally, the comprehensive survey about the explainable methods in machine learning by Vilone and Longo [2020], and Du et al. [2019] also aided in finding the relevant techniques for our problem.

4. Datasets

In this section, we will see the overview of various datasets used for experimentation in the thesis and briefly summarise the exploratory data analysis for each dataset.

4.1 An Overview of Datasets

4.1.1 The Kaggle Dataset

The Fake News dataset[25], made available by Kaggle, is one of the most widely used datasets in research to detect fake news. This had approximately 4000 news articles with 53% (2135) fake and 47% (1870) real articles. The real news articles have been scrapped from well-established sources (based on a study by a famous author J.J. Pryor[26]), namely Reuters, The New York Times, The BBC, and CNN. The fake news articles are from multiple sources, such as daily buzz, which has been flagged unreliable by the USA's fact-checking website Politifact[27]. We were provided with the articles' URL, headline, and body for the analysis. For our analysis, we considered only the headline and the body.

4.1.2 ISOT Fake News Dataset

The University of Victoria provided ISOT Fake News Dataset[28] with two categories, fake and real. The real articles were obtained by crawling the data from a single source, namely, Reuters.com, which is widely accepted as a trustworthy news source. On the other hand, fake news crawled from multiple sources was flagged unreliable by the USA's fact-checking websites Politifact and Wikipedia. Although the articles have different topics of discussion, the main highlight for this dataset is the topics of politics and world news. The dataset has a total of 44183 articles, with 51% fake

[25] https://www.kaggle.com/datasets/jruvika/fake-news-detection

[26] https://jjpryor.medium.com/how-statistically-biased-is-our-news-f28f0fab3cb3

[27] https://www.politifact.com/article/2017/apr/20/politifacts-guide-fake-news-websites-and-what-they/

[28] https://www.uvic.ca/ecs/ece/isot/datasets/fake-news/index.php

articles (23481) and 49% (21417) real articles. Additionally, for every news article, we had a title, text, topic/type, and the date the article was published. Since ISOT wanted to keep the dataset similar to that of Kaggle, they mainly had articles between 2016 and 2017. The articles were made available post-cleaning and processing, apart from a few deliberate mistakes in the fake news articles. For our analysis, we used just the title and the text.

4.1.3 The Fake News Corpus

The Fake News Corpus[29] is an open source data set containing articles from 1001 domains[30]. The domains are associated with particular labels assigned as per the information available via fact-checking agencies. The domains are associated with one or multiple labels: Fake News, Satire, Extreme Bias, Conspiracy Theory, State News, Junk Science, Hate News, Clickbait, Proceed With Caution, Political and Credible. They have provided 9,408,908 news articles from these domains to be used for the research on detecting fake news, mainly using Deep Learning algorithms. The labels of the articles are assigned based on the labels of the domain. The articles were scrapped from the given 1001 domains to extract the article text, title, URL, label, authors, meta keywords, and meta description. The data is provided in its raw format without cleaning and contains various topics such as politics, sports, business, international, and war. Hence, this reduces the Bias in the data set towards a particular topic or the articles from a particular source. However, on the other hand, this increases the challenge of detecting fake news because of the absence of apparent patterns amongst similar topics or articles from a similar domain.

For our analysis, we chose the news under the label *credible* as *Real News* articles (1,920,139), and for the *Fake News* articles, we combined the articles under the label *Fake News* (928,083) and *Extreme Bias* (1,300,444). We combine the articles from two domains to make the dataset more challenging for detection and act as efficient data for testing our pipeline-based approach. According to the Fake News Corpus, credible sources are the ones that are known to publish news articles that follow the traditional and ethical procedures stated in the world of journalism. On the other hand, Fake News is given to domains that publish forged information, circulate/spread deceptive articles, and flagrantly distort factual information and reports. The Extreme Bias-labeled sources are the ones that are known to publish one-sided information, out-of-context content, propaganda, and opinionated comments as facts. For our analysis, we subset approximately 52,000 articles, wherein approximately 26,000 articles are from credible domains, approximately 13,000 articles from the Extreme Bias domains, and approximately 13,000 articles from the Fake News domains.

4.1.4 The Subset of Fake News Corpus for Attribution

This corpus is a subset of the fake news corpus, which was developed to test the performance of the High-Level Descriptor using attribution in a biased environment. The bias is introduced by filtering out the articles containing the keywords *nuclear*

[29] https://github.com/several27/FakeNewsCorpus
[30] http://www.opensources.co/

and *korea*, wherein keywords are case insensitive. For the fake articles, we just used the label *Fake News*. This was done to create bias based on topics being talked about and reduce the challenge by not including the biased news amongst the fake news. We extracted 11963 articles for experimentation, containing 66.17% (7917) real articles and 33.82% (4046) fake articles.

4.1.5 External Datasets

We used two external datasets to support the extraction of specific high-level descriptors: the Multi-Perspective Question Answering (MPQA)[31] and the Myers-Briggs Personality Type (MBTI)[32] Dataset.

The MPQA dataset is a famous opinion corpus containing news articles manually annotated for opinions, beliefs, and sentiments. There are 535 news articles, and we use manually annotated labels for the topics the articles are assigned to. Every news article has been labeled with one or more topics. The topics are from different domains, such as crime, domestic politics, the military, and the environment. It is used as an external dataset to train the classifier to learn the topics as part of a multi-label classification problem.

The MBTI dataset is based on the MBTI indicator, which is a personality system capturing 16 different personality systems (i) Introvert (I)-Extrovert (E) (ii) Intuition (N)-Sensing (S) (iii) Thinking (T)-Feeling (F) (iv) Judging (J)-Perceiving (P). The dataset contains 50 posts from the personality cafe forum by 8675 people. For each person, the label of their personality type is encoded as a four-letter code. For example, the code ISFP indicates Introvert, Sensing, Feeling, and Perceiving personality traits. This dataset was used externally to see the connection between the fake and real texts and personality types. This was also used in a multi-label classification setting to annotate the texts into 4 of 16 personality types.

The statistical summary of all the datasets is given in Table 4.1.

Dataset	Number of Articles	Real Articles	Fake Articles	Focus
Kaggle	4005	1870	2135	Mixed
ISOT	44183	21417	23481	Politics, World
The Fake News Corpus	9,408,908	1,920,139	928,083 (Fake), 1,300,444 (Bias)	Mixed
Subsetted Fake News Corpus	11963	7917	4046	Politics and War

Dataset	Real Sources	Fake Sources
Kaggle	CNN,BBC, Reuters, NY times	Daily Buzz, Before it's News etc.
ISOT	Reuters	Flagged by PolitiFact
The Fake News Corpus	Reuters, BBC, etc.	Before it's News, ABC News, American Lookout, etc.
Subsetted Fake News Corpus	Reuters, BBC, etc.	Before it's News, ABC News, American Lookout, etc.

Table 4.1: The table shows the summary of the datasets used in the Thesis. We used only a subset of data for the Fake News Corpus for the analysis, whereas all the other datasets were used in their entirety.

[31] https://mpqa.cs.pitt.edu/corpora/mpqa_corpus/
[32] https://www.kaggle.com/datasets/datasnaek/mbti-type

4.2 Exploratory Data Analysis (EDA)

Before extracting the features from the datasets, we analyzed each to get an overview of the data in terms of its textual elements. For uniformity, we have considered exploring five main aspects across all the datasets, namely (i) Word Cloud Analysis (ii) Unigram and Bigram Level Analysis (iii) Analysis of Length of the Text (iv) Analysis of Grammatical Mistakes (v) Topic Analysis. We will talk about them in the following subsections.

4.2.1 The Kaggle Dataset

4.2.1.1 Word Cloud Analysis

For the word cloud analysis, we aimed to see what the overall word usage of the dataset looks like and then to analyze if the word usage differs across the fake and real news articles. Overall, the words *game*, *trump*, *new*, *year* and *time* have been used the most. Moreover, they all belong to different domains, showing that the dataset talked about diverse topics rather than focusing on the news from a single topic(cf. Figure 4.1a).

Moving on to the results for the fake news articles, we see in Figure 4.1b that fake content is dominated by words like *game*, *team* and *season*. These words have a strong affinity for the sports domain. On the contrary, as seen in Figure 4.5b, the words *president*, *trump*, *world* and *people* are the ones used majorly in real news articles, which have a solid affinity for politics. This analysis succeeded in hinting at the types of techniques used to detect fake news at the word level.

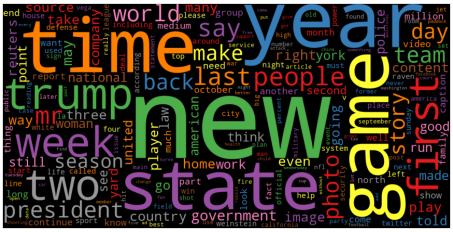
4.2.1.2 Unigram and Bigram Level Analysis

After analyzing the type of word usage, we wanted to see if there were similarities or differences in the frequency of the top used words in the article's content. We analyzed this trend at the unigram and bigram levels. At the unigram level, the word *trump* occurs in fake and real articles but with a higher frequency in the real articles. For the fake texts, the frequency of the words such as *game*, *season*, and *play* is relatively high and also unique to the fake texts. On the other hand, terms like *time*, *story* and *government* contribute to the usage and uniqueness of the real texts (cf. Figure 4.2).

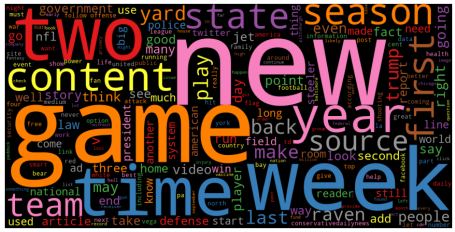
Comparing the Bigrams, we see that the word *Donald Trump* occurs in real and fake texts. This can be attributed to the popularity of news content related to him. The Bigram *think Fact* is an exciting find as it can hint toward fake news overly using the word *Fact* to prove its authenticity. On the other hand, the content of the real news articles was dominated by the bigrams *United States* and *image copyright*. Using copyright can be seen as proof to validate the article's content (cf. Figure 4.3). Based on this preliminary analysis, we decided to check whether the word statistics had an essential role in differentiating between fake and real articles.

4.2.1.3 Analysis of the Length of the Text

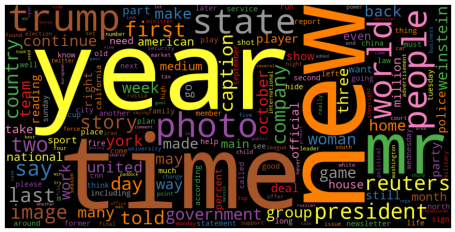
The following analysis caters to checking the affinity of the real and fake articles towards the length of the content. First, we see that for real and fake articles, we have



(a) The figure depicts the word cloud for the overall texts present in the Kaggle dataset.



(b) The figure depicts the word cloud for the fake texts present in the Kaggle dataset.



(c) The figure depicts the word cloud for the real texts present in the Kaggle dataset.

Figure 4.1: The figures present an overview of the word usage over the complete, fake, and real texts in the Kaggle dataset. They are used to get a brief outline of what kind of words are used to describe a particular category.

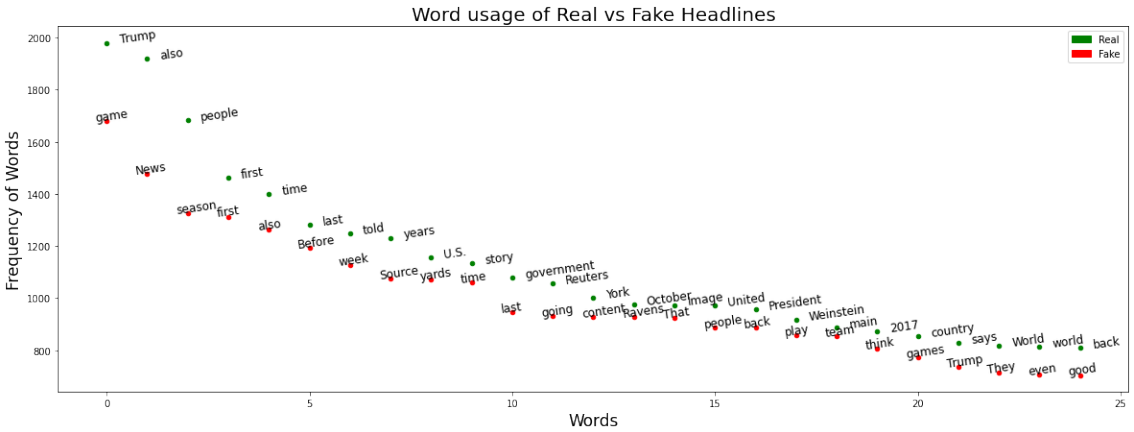


Figure 4.2: The figure shows the comparison between the frequency of 25 most commonly used words in the real and fake texts of the Kaggle dataset.

an almost similar distribution of articles in the lower length bins, i.e., between 0 and 450. However, post the length of 450, the real articles are more uniformly distributed across the lengths 450-750, whereas the fake articles tend to be primarily binned in the shorter length bins (cf. Figure 4.4). This characteristic can be attributed to the fact that fake articles mostly leave out the facts for the stated reports and are prone to shorter texts.

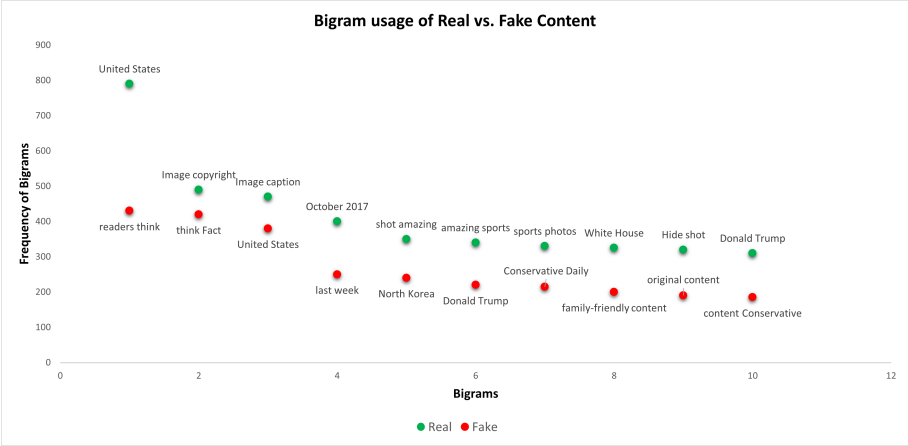
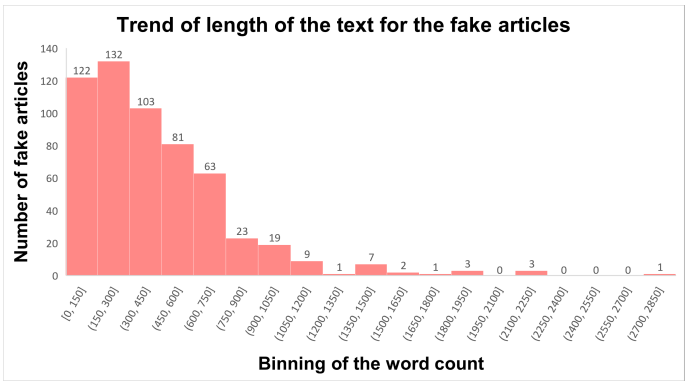
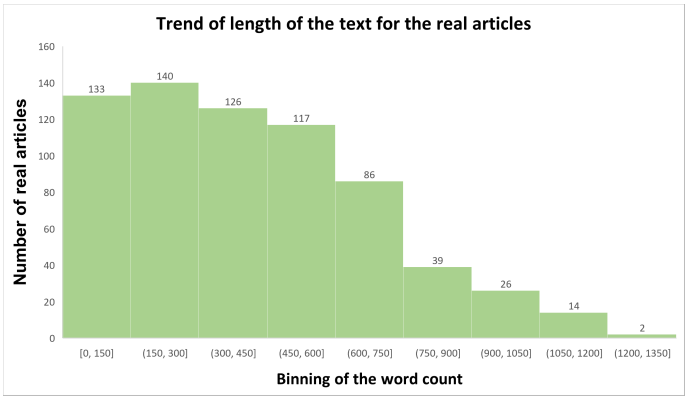


Figure 4.3: The figure shows the comparison between the frequency of 10 most commonly used bigrams in the real and fake texts of the Kaggle dataset.



(a) The sub-figure shows how the fake articles are distributed across the length of the content.



(b) The sub-figure shows how the real articles are distributed across lengths of the content.

Figure 4.4: The figure shows the comparison between the distribution of the articles across varying length bins in the Kaggle dataset. The analysis is done to determine if there are any patterns related to the content's size that can help distinguish between fake and real news.

4.2.1.4 Analysis of Grammatical Mistakes

Taking a step deeper into lexical analysis, we also explored the types of grammatical mistakes committed in the content. For the fake articles (cf. Figure 4.5a), typos entirely overpower the grammatical mistakes. Typos here are the spelling mistakes in the content, which might be the case because fake news is often not reviewed before publishing. The real news, on the other hand, has both typos and typography errors, meaning problems with the spellings and usage of capitalization and lower case letters (cf. Figure 4.5b). To summarise, we saw that both kinds of texts had a fair number of grammatical errors, and no particular category stood out for either text.

4.2.1.5 Topic Analysis

To move toward the direction of latent characteristics, we did a preliminary analysis using topic modeling to find the types of topics being talked about in the text. For this, we did not tune the parameters except for the number of topics, as it was done only to get a rough idea of the topics being talked about. For the Kaggle dataset we found 3 distinct topics (cf. Figure 4.6a). The fact that we had non-overlapping circles, as shown by the visualization by the state-of-the-art visualization tool for topic modeling, *pyLDAvis*[33], gives a qualitative measure of the topics being built (cf. Sievert and Shirley [2014]).

Going through the top 15 words being talked about in each topic (cf. Figure4.6b), the three main topics that came across were *Politics* (state, trump, government), *Sports* (game, season, team) and *Lifestyle* (people, new, time). This also agrees with our word-level analysis and the factual information about the dataset talking about mixed domains.

4.2.2 ISOT Fake News Dataset

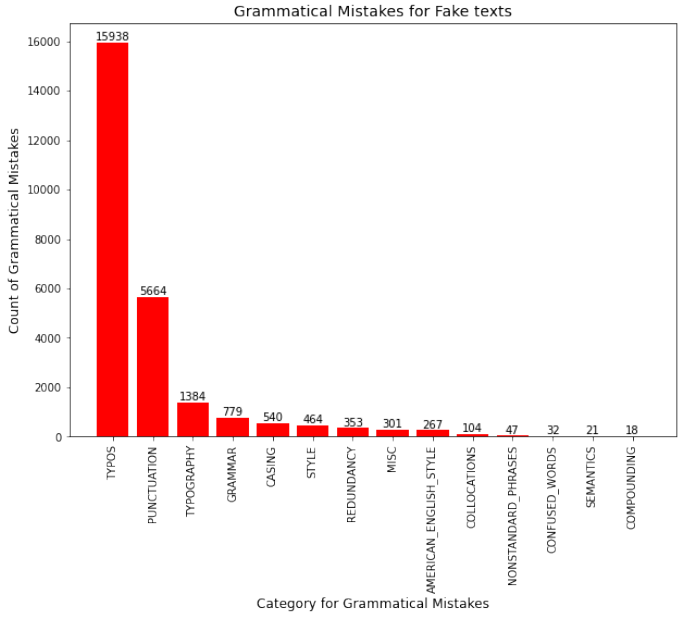
4.2.2.1 Word Cloud Analysis

For the ISOT dataset, the overall word cloud is mostly overpowered with the words from the domain of the politics, such as *trump*, *election*, *president* and *republican* (cf. Figure4.7a). Therefore, the real and the fake word clouds of the ISOT dataset also are very similar since the news from the political domain overpowers them. This is also in agreement with the factual information of the ISOT dataset stating the content to be mostly talking about politics and world news. Hence just analyzing the words used would not be sufficient for this dataset to discriminate between the real and fake texts.

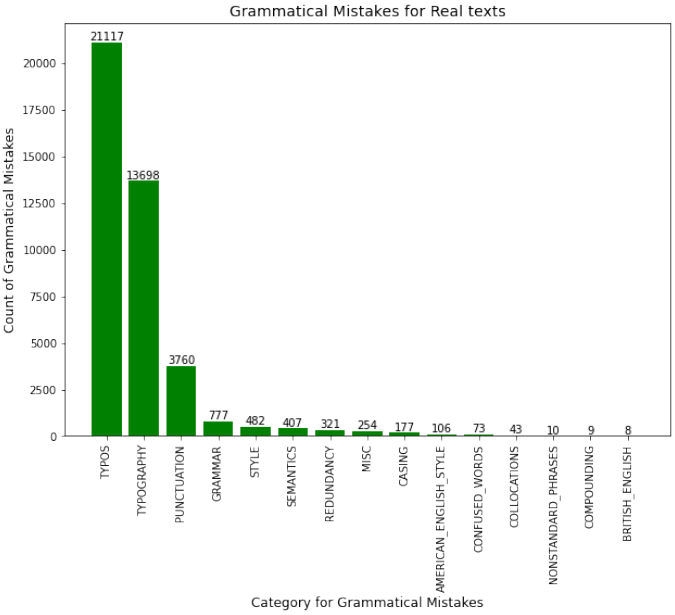
4.2.2.2 Unigram and Bigram Level Analysis

For the unigram analysis, we saw that the frequency of most commonly used words in the fake and real texts is very similar (overlapping). For example, the word *Trump* is the most frequently used word in both the texts, and even the words overlap, such as*America*, *campaign* and *President*. Hence nothing substantial can be said from the unigram analysis of the ISOT dataset (cf. Figure 4.8).

[33] https://pypi.org/project/pyLDAvis/



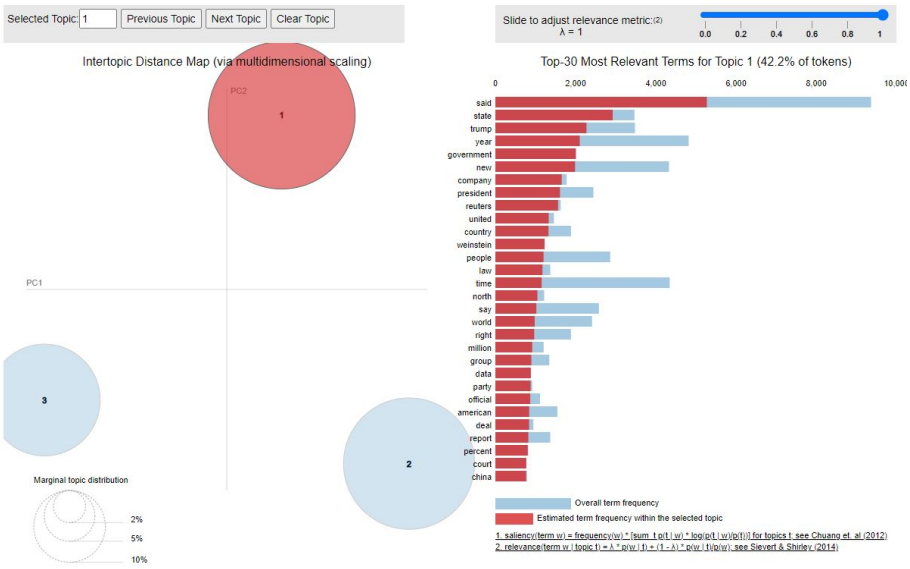
(a) The subfigure shows the frequency of different types of grammatical mistakes occurring in the fake texts.



(b) The subfigure shows the frequency of different types of grammatical mistakes occurring in real texts.

Figure 4.5: The figure shows a comparison of different types of grammatical errors across the fake and real articles. This analysis was done to see if there are occurrences of certain kinds of grammatical mistakes across different categories.

On the Bigram front, also we see similar trends wherein both the fake and real news have *Donald Trump* and *White House* occurring very frequently (cf. Figure 4.9). The only important information from this analysis is that the content is primarily political; hence, we could use domain-specific features to differentiate between fake and real texts.



(a) The figure shows the topics being discussed in the Kaggle Dataset. On the left side of the figure, the circles indicate the topics, and on the right side, we have a representation of the top 30 relevant terms for each topic.

	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8	Word 9	Word 10	Word 11	Word 12	Word 13	Word 14
Topic 0	said	state	trump	year	government	new	company	president	reuters	united	country	weinstein	people	law	time
Topic 1	game	week	season	team	play	player	yard	time	year	raven	league	win	content	run	photo
Topic 2	said	time	like	people	year	new	news	story	image	say	woman	trump	http	com	day

(b) The subfigure shows the top 15 words used for each topic. It is a more concise representation of the topic represented in Figure 4.6a.

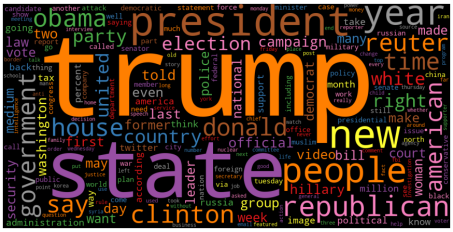
Figure 4.6: The figure demonstrates the topics discussed in the Kaggle dataset using the pyLDAvis tool. The purpose of this analysis was to get a hint of latent characteristics of the dataset to see if multiple topics are being talked about or if the content talks about similar things.

4.2.2.3 Analysis of the Length of the Text

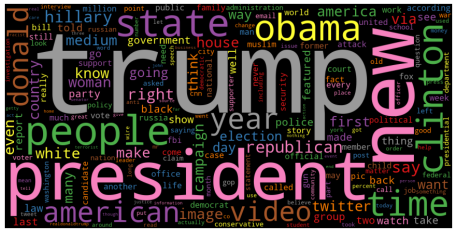
In terms of analysis in terms of the length of the content of the text, we see that for the fake texts, there is a very uniform distribution in the count of news sharing the lengths between $0 - 150, 150 - 300$ and $300 - 450$ (cf. Figure 4.10a). For the real news, this uniformity also extends to $450 - 600$ and $600 - 750$ (cf. Figure 4.10b). Like the Kaggle dataset, we see the affinity of the fake news towards smaller content length.

4.2.2.4 Analysis of Grammatical Mistakes

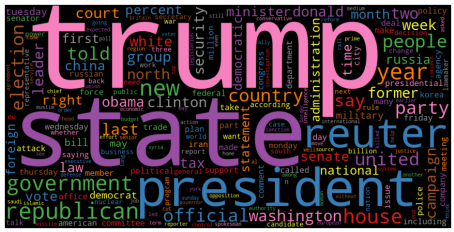
For the grammatical mistakes, both the fake and real content has mostly Typographical errors and Typos, which is similar to the findings in the Kaggle dataset. The primary point of difference is in terms of errors related to punctuation, grammar, casing, and redundancy. The punctuation mistakes are primarily present in the real texts, whereas fake texts have more grammatical, casing, and redundancy problems (cf. Figure 4.11). Although the top grammatical mistakes did not help in differentiating between the different categories, there were other categories, such



(a) The figure depicts the word cloud for over the overall texts present in the ISOT dataset.



(b) The figure depicts the word cloud for the over fake texts present in the ISOT dataset.



(c) The figure depicts the word cloud over the real texts present in the ISOT dataset.

Figure 4.7: The figures present an overview of the word usage over the complete, fake, and real texts in the ISOT dataset. They are used to get a brief outline of what kind of words are used to describe a particular category.

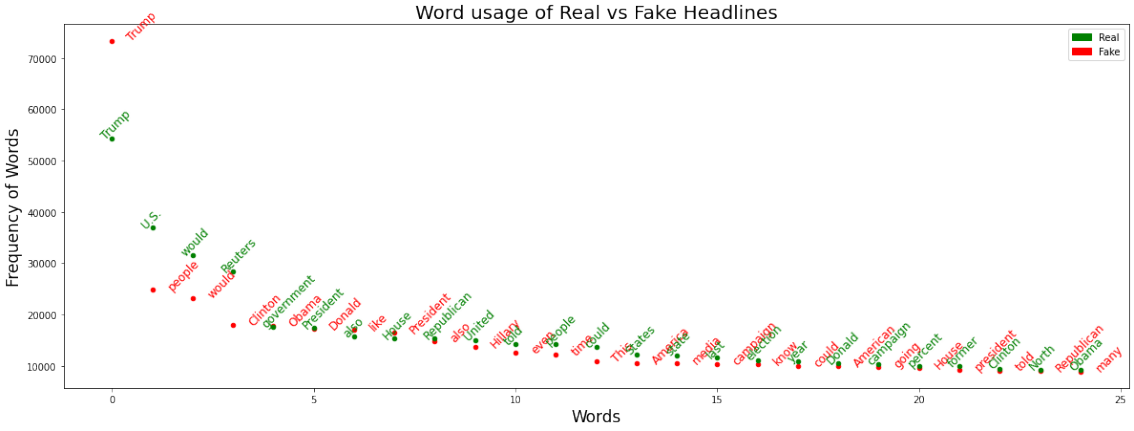


Figure 4.8: The figure shows the comparison between the frequency of 25 most commonly used words in the real and the fake texts of the ISOT dataset.

as punctuations and redundancy, which motivated us to explore them as features further.

4.2.2.5 Topic Analysis

From all the preliminary analysis done on the ISOT dataset, we saw that the content is overpowered by political news. We wanted to check if LDA can capture the same information or not. We set the number of topics to 3 initially for experimentation. We see that although we get three non-overlapping bubbles (cf. Figure 4.12a), all of them talk about the words with an affinity towards politics (cf. Figure 4.12b).

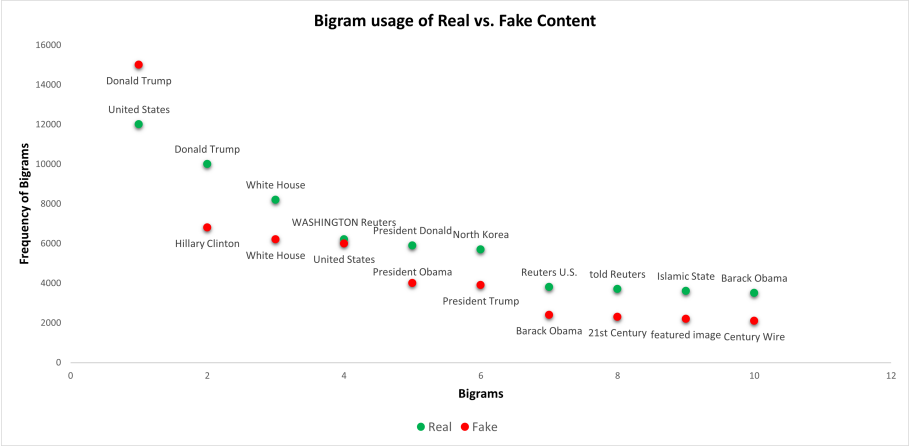
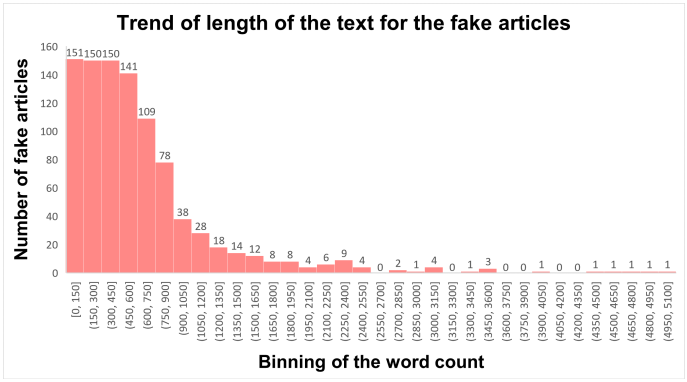
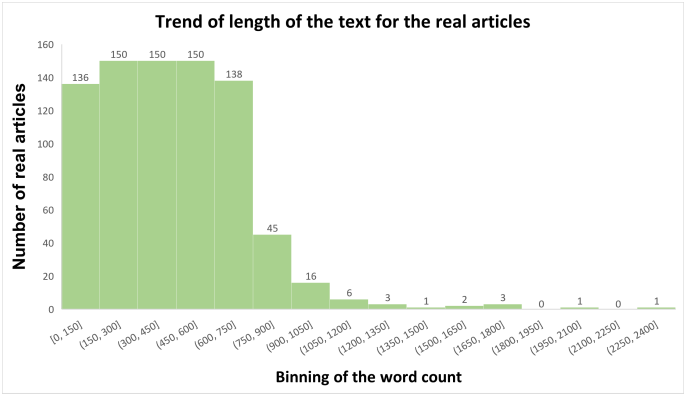


Figure 4.9: The figure shows the comparison between the frequency of 10 most commonly used Bigrams in the real and the fake texts of the ISOT dataset.



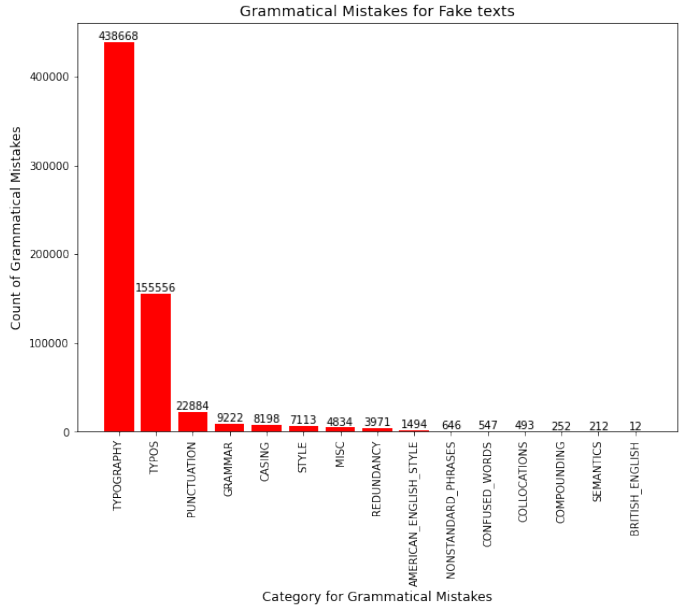
(a) The subfigure shows how the fake articles are distributed across lengths of the content.



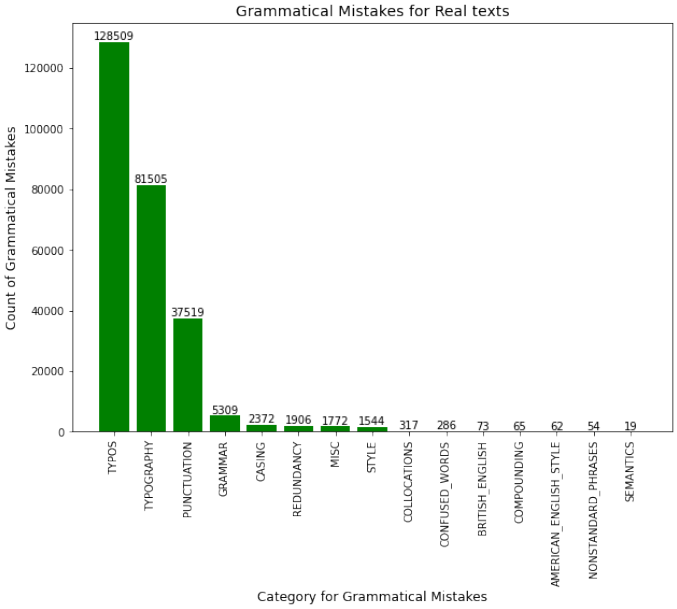
(b) The subfigure shows how the real articles are distributed across lengths of the content.

Figure 4.10: The figure shows the comparison between the distribution of the articles across varying length bins in the ISOT dataset. The analysis is done to determine if there are any patterns related to the size of the content that can help distinguish between fake and real news.

For instance, Topic 0 talks about trump and Clinton, whereas Topic 1 talks about Russia, government, and security which also has an affinity to world news.



(a) The subfigure shows the frequency of different types of grammatical mistakes occurring in the fake texts.



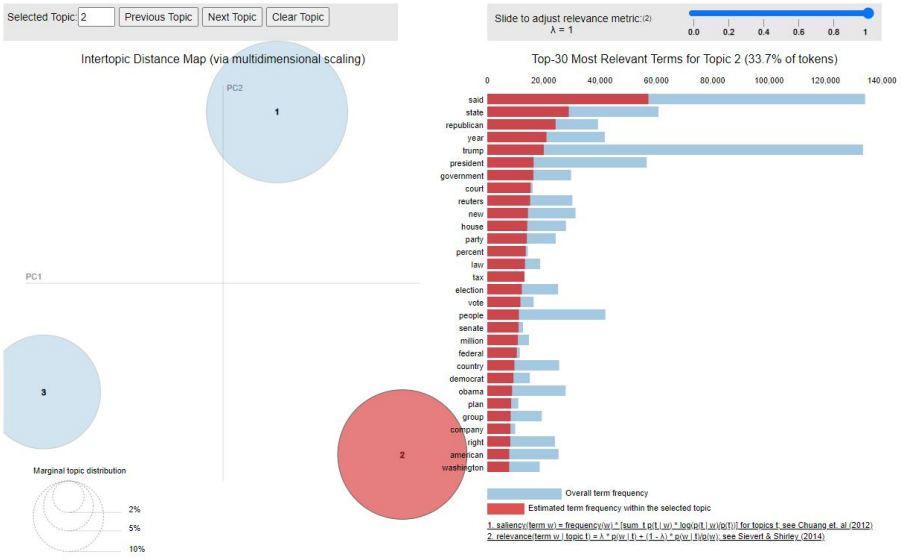
(b) The subfigure shows the frequency of different types of grammatical mistakes occurring in the real texts.

Figure 4.11: The figure shows a comparison of different types of grammatical errors across the fake and real articles. This analysis was done to see if there are occurrences of certain kinds of grammatical mistakes across different categories.

4.2.3 The Fake News Corpus

4.2.3.1 Word Cloud Analysis

The overall word cloud of the Fake News Corpus shows the usage of words such as *time*, *state*, *people*, *president* and *million*. All of the words belong to a very different domain of topics which comes from the fact that the dataset was created without



(a) The figure shows the topics being discussed in the ISOT Dataset. On the left side of the figure, the circles indicate the topics, and on the right side, we have a representation of the top 30 relevant terms for each topic.

	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8	Word 9	Word 10	Word 11	Word 12	Word 13	Word 14
Topic 0	trump	clinton	people	president	said	like	donald	time	hillary	american	republican	news	campaign	obama	white
Topic 1	said	state	trump	president	reuters	official	united	russia	government	security	russian	military	north	china	told
Topic 2	said	state	republican	year	trump	president	government	court	reuters	new	house	party	percent	law	tax

(b) The subfigure shows the top 15 words used for each topic. It is a more concise representation of the topic represented in Figure 4.12a

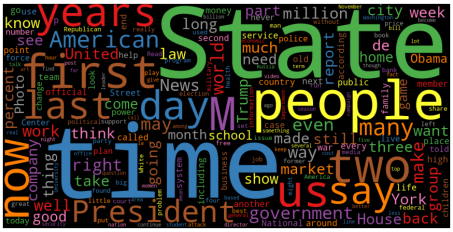
Figure 4.12: The figure demonstrates the topics being talked about in the ISOT dataset using the pyLDAvis tool. The purpose of this analysis was to get a hint of latent characteristics of the dataset to see if multiple topics are being talked about or if the content talks about similar things.

any bias toward a specific type of news. For the fake texts, we saw the dominance of words such as *President*, *government* and *market*. For the real texts, the words *say*, *time* and *state* dominated. There was no specific pattern found when it came to word usage (cf. Figure 4.13).

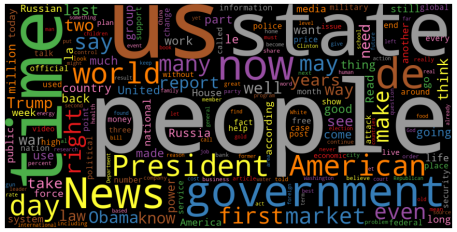
4.2.3.2 Unigram and Bigram Level Analysis

The unigram analysis did not yield significant differences for the Fake news corpus. Both the categories made use of words such as *people*, *government* and *also*. However, the word *government* was more frequent in the fake news than in the real one, which can hint at the fact that fake news seems to be a popular choice when it comes to governmental news topics. There was also a uniqueness of the word *market* in the content of fake news text (cf. Figure 4.14).

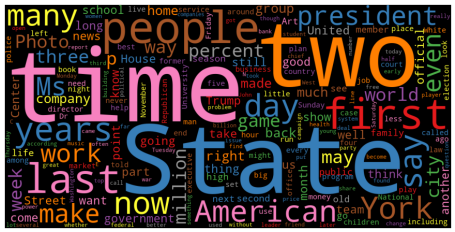
On the Bigram front, there was a substantial overlap of the bigrams and their frequencies in the texts. For example, the word *White House*, *Donald Trump*, *Barack Obama* and *Hillary Clinton* is used approximately the same number of times in both fake and real texts (cf. Figure 4.15). This indicated that word-based statistics would not be enough to differentiate the texts in the fake news corpus, and we would also have to explore the text's semantics.



(a) The figure depicts the word cloud the over the overall texts present in the Fake News Corpus.



(b) The figure depicts the word cloud for the over the fake texts present in the Fake News Corpus.



(c) The figure depicts the word cloud over the real texts present in the Fake News Corpus.

Figure 4.13: The figures present an overview of the word usage over the complete, fake, and real texts in the Fake News Corpus. They are used to get a brief outline of what kind of words are used to describe a particular category.

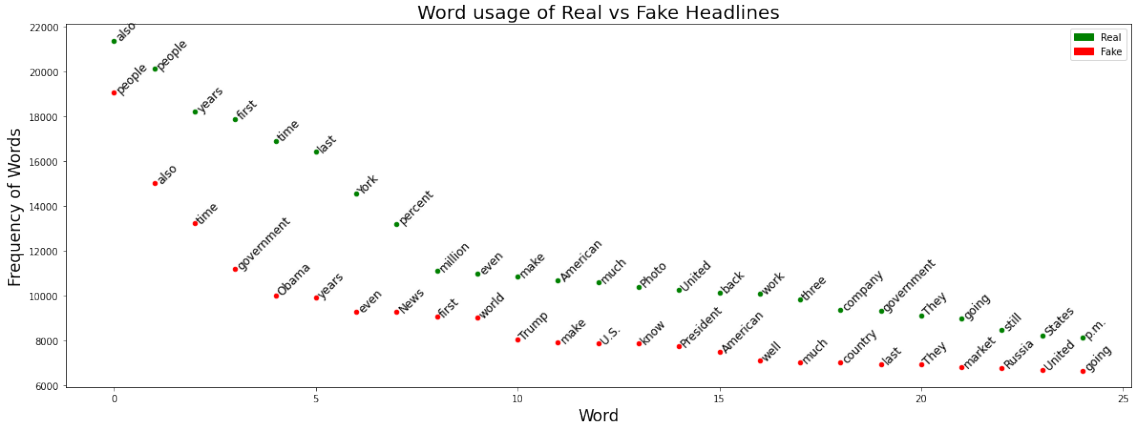


Figure 4.14: The figure shows the comparison between the frequency of 25 most commonly used words in the real and the fake texts of the Fake News Corpus.

4.2.3.3 Analysis of the Length of the Text

Unlike the ISOT and Kaggle dataset, no pattern was observed for the Fake News Corpus in terms of the length of the news content. Both the groups had approximately similar distribution across all the bins. Both the real and the fake texts have an affinity toward shorter content length (cf. Figure 4.16). This, along with the previous analysis, motivates us to look into semantical and style-based features for this dataset.

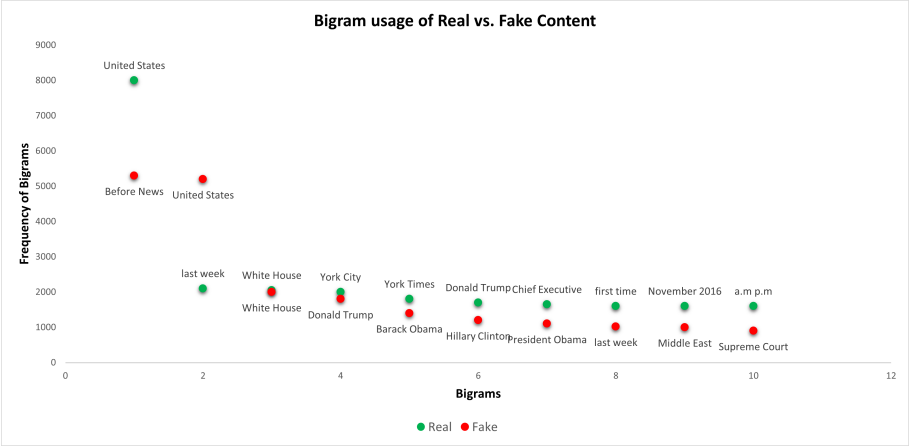
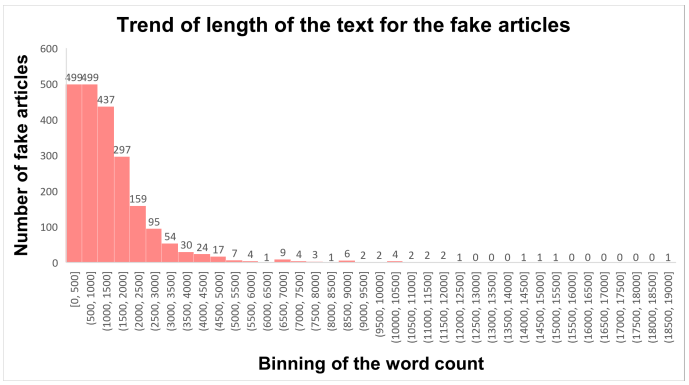
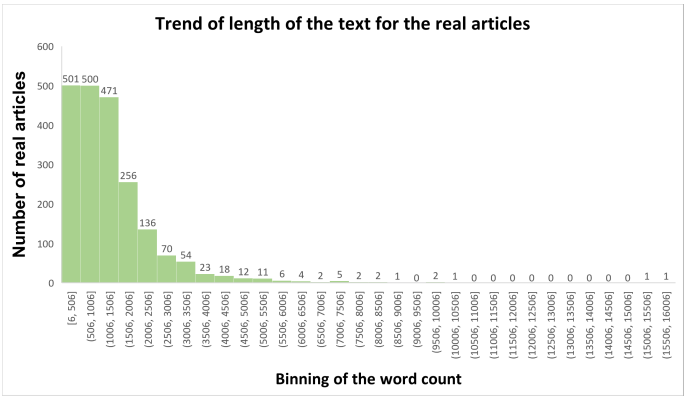


Figure 4.15: The figure shows the comparison between the frequency of 10 most commonly used Bigrams in the real and the fake texts of the Fake News Corpus.



(a) The subfigure shows how the fake articles are distributed across lengths of the content.

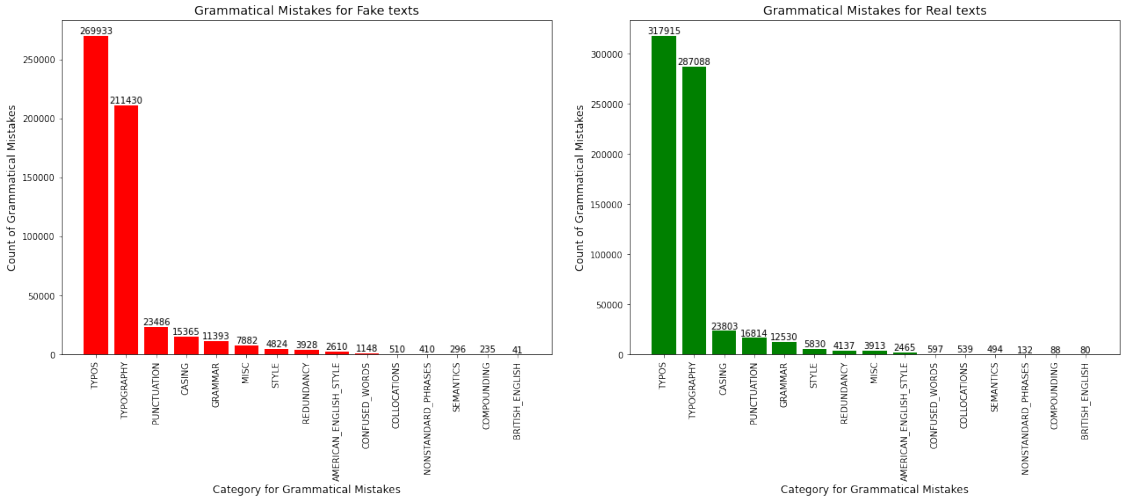


(b) The subfigure shows how the real articles are distributed across lengths of the content.

Figure 4.16: The figure shows the comparison between the distribution of the articles across varying length bins in the Fake News Corpus. The analysis is done to determine if there are any patterns related to the size of the content that can help distinguish between fake and real news.

4.2.3.4 Analysis of Grammatical Mistakes

On the front of grammatical errors, just like the Kaggle and the ISOT dataset, Typos and Typographical errors are the ones found the maximum times for both real and fake texts. The difference was in terms of punctuation and casing-related errors, where the former were primarily found in the fake texts, whereas the latter dominated the real texts (cf. Figure 4.17). The differences in the types of grammatical errors motivated us to explore style-based featrues for this dataset.



(a) The subfigure shows the frequency of different types of grammatical mistakes occurring in the fake texts.

(b) The subfigure shows the frequency of different types of grammatical mistakes occurring in the real texts.

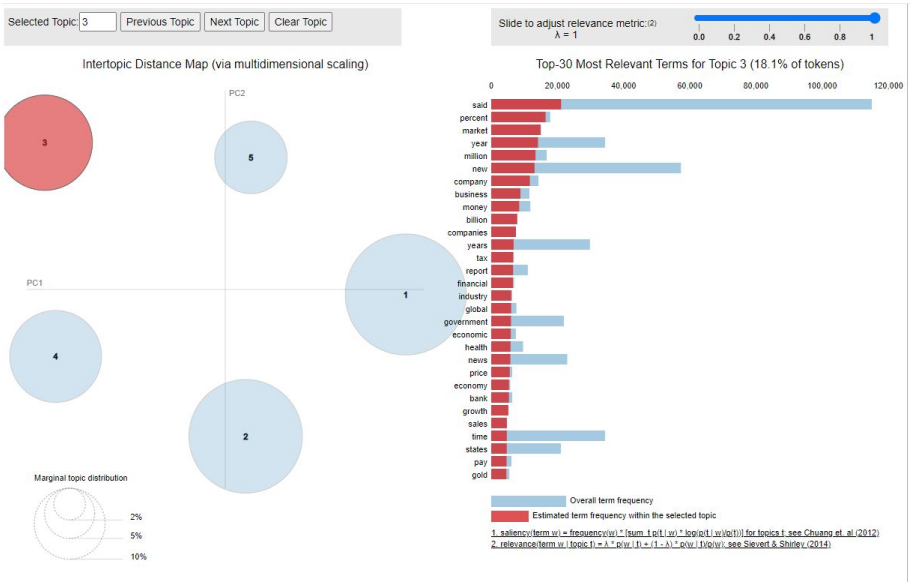
Figure 4.17: The figure shows a comparison of different types of grammatical errors across the fake and real articles. This analysis was done to see if there are occurrences of certain kinds of grammatical mistakes across different categories.

4.2.3.5 Topic Analysis

As we saw in Figure 4.13a, the corpus had word usage across different domains. We wanted to confirm this finding via topic modeling. LDA indicated 5 major topics present in the corpus (cf. Figure 4.18a) wherein every topic had words from different domain (cf. Figure 4.18b).For example, Topic 4 talks about words having affinity to the Economics domain with words such as *market*, *company*, *money* and *business*. Where Topic 2 talks mainly about Politics with the words like *president*, *government* and *Obama*. This showed that, unlike the ISOT dataset, domain-specific features would not make much sense here, and we would have to focus on the domain agnostic feature extraction.

4.2.4 Summary

From the EDA done over the datasets, we saw that the Kaggle dataset showed differences between the fake and real texts in terms of word usage, wherein the fake texts were dominated by words from *sports* domain. In contrast, the real texts affinity towards *politics*. The texts are also different in terms of length, unigram, and



(a) The figure shows the topics being discussed in the Fake News Corpus. On the left side of the figure, the circles indicate the topics, and on the right side, we have a representation of the top 30 relevant terms for each topic.

	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8	Word 9	Word 10	Word 11	Word 12	Word 13	Word 14
Topic 0	said	new	york	game	year	street	season	photo	team	city	art	play	music	center	time
Topic 1	said	water	new	use	food	north	people	used	air	space	energy	area	china	talk	like
Topic 2	said	president	state	government	trump	states	people	obama	united	police	american	war	political	new	law
Topic 3	like	people	just	time	said	know	did	life	new	years	think	way	does	world	make
Topic 4	said	percent	market	year	million	new	company	business	money	billion	companies	years	tax	report	financial

(b) The subfigure shows the top 15 words used for each topic. It is a more concise representation of the topic represented in Figure 4.18a.

Figure 4.18: The figure demonstrates the topics discussed in the Fake News Corpus using the pyLDAvis tool. The purpose of this analysis was to get a hint of latent characteristics of the dataset to see if multiple topics are being talked about or if the content talks about similar things.

bigram usage. Though there was mixed news, the content was mainly about *sports* and *politics*.

For the ISOT dataset, we had no significant differences when it came to when analyzing the word, unigram, and bigram usage. However, the fake texts in the ISOT dataset showed affinity toward shorter lengths of content. Moreover, the ISOT dataset was dominated by most of the content from the *political* domain.

There were no significant differences in word usage, unigram-bigram usage, and content length for the Fake News Corpus. However, there were differences in the grammatical mistakes found in either category, wherein *puntuation* mistakes were primarily found in the fake texts, and *casing* errors were seen for the real texts. Furthermore, we saw that the content was not dominated by the news from a single domain but had varying aspects such as *economics*,*politics*,*sports* etc.

The quantitative summary of all the three datasets is described in Table 4.2. We observe that it might be challenging to differentiate the fake and real news in the Fake News Corpus solely based on the words.

	Average length			Average number of grammatical mistakes	
Dataset	Overall	Fake News	Real News	Fake News	Real News
Kaggle	282	229	341	12	22
ISOT	234	238	257	27	12
The Fake News Corpus	461	438	480	23	23

Table 4.2: The table shows the quantitative summary of the length of the content and the grammatical mistakes per article in the three datasets. We see that the Fake News Corpus shows few differences in terms of the word-based features.

5. Concept for FACADE

As discussed in the Introduction, the volume of news that has to be checked every day (and therefore every second) is exceptionally high. Due to that, it is necessary to create a tool that solves the problems mentioned in Section 1.1 and also processes this high volume of data. Furthermore, due to our preliminary analysis of the data set, we assume that certain types of news are easier to distinguish as fake compared to others. This is why we design FACADE as a cascading system. A cascade hereby consists of three phases:

1. **Feature Extraction:** Features from the news are extracted that are used for the subsequent classification.
2. **Classification:** A classifier is applied to news using the extracted features to give a probability of how likely the news is fake or real.
3. **Filtering:** News are filtered based on their probability to distinguish between the most likely real or fake news according to the current classifier and those that need further classification. The goal of the filtering is to reach a certain accuracy on the confidently classified news, which we denote as A_{min}^{conf}.

The news left over after filtering, i.e., those that might need further classification would go into the next level of the cascade. Each cascade we are calling a *pipeline*. For our thesis, we designed FACADE only to have two cascades and, therefore, two pipelines. This would be easily extendable, though, for a more detailed approach. Our two pipelines consist of the following:

1. **Low-Level Descriptors (LLD):** The first pipeline extracts basic statistical features from the text and headlines of the articles to be classified.
2. **High-Level Descriptors (HLD):** The second pipeline uses more difficult and time-consuming features such as sentiments, entailments, attribution, syntactical structure, tones, and latent topics.

In the end, the output of the pipelines is explained, whereas the assumption that a decision is best explained by the most important features used for distinguishing the news article as fake or not. These features should optimally be already explainable by themselves. The complete system is depicted in Figure 5.1.

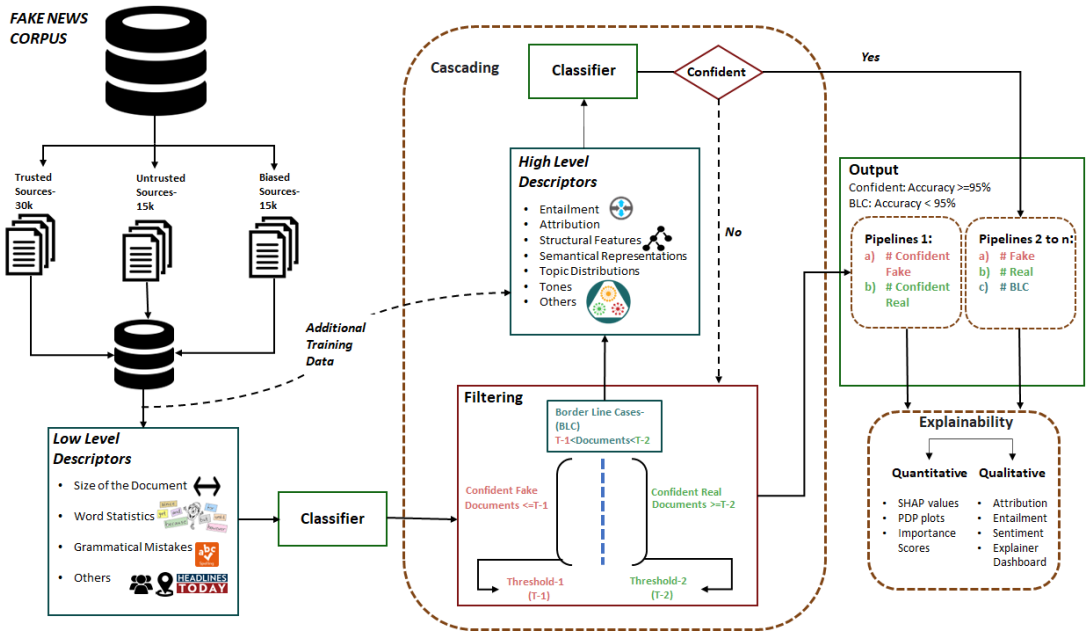


Figure 5.1: The diagram depicts our system FACADE, which is composed of three different sections. The first section on the left depicts the data representation and the extraction of low-level descriptors like statistical features and classification of those. The second section in the middle contains the cascading process, which first filters documents based on confidence and classifies the not-confident documents using higher complexity features. The final component on the right explains the decisions made within the pipeline to classify an article as fake or real.

The figure depicts the dataset as an input to the first cascade, whose role will be discussed in the following section.

5.1 Cascading

5.1.1 Data Representation

Our system is designed to solve a binary classification problem. Therefore the data was transformed accordingly. First, the articles from various news sources belonging to specific categories such as biased, fake, and real news were considered. Next, multiple sources were used to reduce bias toward a specific topic, and multiple categories were then narrowed down to *fake* and *real* to increase the problem's complexity. Finally, the known fake and biased sources were merged into the *Fake* category, whereas articles from multiple known reliable sources were used for the *Real* category. The problem was designed as a balanced dataset problem, with an equal number of articles in each category. The datasets explored and used are described in detail in Section 4. The articles and their headlines are further crafted into custom

features for each pipeline, fed to their respective classifiers. Furthermore, the articles are either classified or sent to the cascading steps for further research based on the confidence of predictions. We need to feed the articles their headlines in raw string format for FACADE to kick in.

5.1.2 Feature Extraction

Different types of features are extracted from the prepared dataset. We distinguish between the following types of features:

1. Low-Level Descriptors (LLD): We create basic statistical features from the text and headlines of the articles to be classified. The idea behind having different pipelines based on complexity was that not all news is difficult to unmask, and those characteristics such as document size, grammatical errors, and other statistical features are sufficient to identify fake articles. This, in turn, reduces computational power and speeds up the classification process. Furthermore, because of the simplistic and realistic nature of the features, they also lead to implicit explainability.
2. Standalone High-Level Descriptors (S-HLD): For very well-written fake news, we then extract the semantically meaningful features, which are high in complexity and make use of extra preparatory steps like clustering and text segmentation, which will be described in Section 5.3. The features include tones, the syntactic structure using parts of speech (POS), topics, WordNet's hypernym-hyponym structure, sentiments, entailment, personality, and attribution to known fake and authentic sources. These descriptors are partially self-explanatory features.
3. Combined High-Level Descriptors (C-HLD): The standalone features were then combined to make use of the possibility that some semantical meaning is correlated. Features detecting topics followed by attribution/entailment and a combination of attribution followed by the type of entailment are extracted here. For more details, see Section 5.5.2.
4. Transformed Standalone High-Level Descriptors (T-HLD) : Here, we transform the features extracted in 2 to a more complex form, which in itself is not explainable but has more discriminatory powers. Here we are making custom embeddings using features from both 2 and 3; for example, we make custom embeddings using categories of noun, verb, and adjectives derived from word net metadata information which can be seen in detail in Section 5.5.1

5.1.3 Classification

For binary classifiers, the decision between the classes is made based on the thresholding criteria that are in place. In general, such classifiers use a single threshold system with a lower bound of 0 and an upper bound of 1. Most classifiers have a default setting t of 0.5, which means that all instances (articles in our case) classified with a value less than 0.5 falls into one class (fake in our case), and those classified with a value more excellent than 0.5 falls into another class (real in our case). Instead of using a single threshold t to make decisions, we use a dual-threshold system t_1

and t_2, one for the fake and one for the real classes, respectively. We further classified the correct articles in which the classifier returns a value between 0 and t_1 as confidently fake and a value between t_2 and one as confidently real. The remaining articles, classified with values ranging between t_1 and t_2, are borderline cases (BLC). BLC, along with the incorrectly classified articles, are then moved to the following pipelines, where high-level descriptors based on the semantic structure of the articles are extracted to further classify them as fake or real. We experimented with various thresholding techniques to determine the values of t_1 and t_2. We go over those results in Section 7.1.3.

5.1.4 Filtering

In order to be sure that the predictions made by the classifier are confident enough, as mentioned above, t_1 and t_2 as thresholds for the classifier to distinguish fake and real articles, respectively, have to be defined. We made use of two different methods in order to find these thresholds, which are as follows:

1. **Baseline:** For the baseline method, we calculated the average probability values of the correctly classified fake and real articles and then used these average values as t_1 and t_2.
2. **Dynamic Threshold:** For the dynamic thresholds, we devised the optimization function ($O(t)$) that makes use of both the accuracy (α) and a number of articles that have been confidently classified after threshold (β). Accuracy provides the measure for goodness of fitness of the model wherein the number of confident articles acts as the measure of generalization wherein we will penalize the thresholds that result in more borderline cases. In the end, both parameters ensure that the resulting thresholds are such that there is a balance between A_{min}^{conf} and borderline cases. The optimization function is designed as given in Eq. 5.1. We have further assigned weights w_0 and w_1 to α and β respectively, to control the importance given to each parameter in finding the thresholds.

$$O(t) = w_0\alpha + w_1\beta \tag{5.1}$$

Once we have the values of thresholds in place, we label every article with predicted by the classifier with the probability $< t_1$ as confidently fake and articles predicted with probability $\geq t_2$ as confidently real. Every article predicted with the probabilities between t_1 and t_2 are then termed as Border Line Cases (BLC) and send to the second phase of the pipeline for further investigation.

5.2 Low-Level Descriptors

The next step is to extract the features from the raw text after the data has been appropriately formatted. The main idea is to create self-explanatory features. We assume that most news articles are easily distinguishable between real and fake, implying that it is easier to extract the characteristics of both groups using basic syntactical features. We begin our pipeline by extracting such features to

filter out most articles without consuming much computation power while ensuring explainability at the training level.

The number of words, sentences, grammatical errors, readability, sentiments at the lexical level, parts of speech usage, TF-IDF values, and other factors are used to distinguish between real and fake articles. However, there was a small subset of features that could unearth the characteristic differences at the syntactical level, and they are divided into categories that are discussed as follows:

1. **Size based features:** In this category, we extracted features that defined the size of the article's body and headlines. The assumption is that the sizes of the two articles may differ due to differences in how the content is articulated. Fake news that is poorly written may not care about the explanations of a statement and may lack words. We analyzed these assumptions in detail in Section 4.2. The length of the body and headline, the number of unique words in the body and headline, the average word usage in the body and headline, the type-token ratio, the number of sentences, and the average number of words in the sentences were the main features included in this category.
2. **Syntax and design-based features:** The primary assumption for the design-based features was that the fake news content might use superficial stylistic characteristics to catch the reader's attention. Furthermore, because most fake news articles lack proofreading, it is assumed that they will contain more grammatical errors. Keeping these assumptions in mind, the main features used in this category are the number of grammatical errors, ease of reading scores, the use of different parts of speech such as pronouns, adjectives, etc., the use of special characters, upper case words, stop words, punctuation, and the number of *PERSON* entities in the text.
3. **Term Frequency — Inverse Document Frequency:** Lastly, it is seen that there are certain kinds of words that are commonly used in each kind of article. To use the word usage's power to find the structural differences between real and fake articles, we use Term Frequency — Inverse Document Frequency (TF-IDF). The idea behind TF-IDF is to give more weight to less commonly used words and less weight to more commonly used words. This is because those rarely occurring words are seen to be the most important to the article's content.

5.3 Pre-Processing for High-Level Descriptors

In order to make use of the total extent of high-level descriptors that, for example, capture the semantics of a text, we use text segmentation algorithms to split a news article into smaller chunks and cluster articles to detect meaningful groups inside the news corpus. In contrast, each group might be handled slightly differently and more effectively.

5.3.1 Text Segmentation

For the LLD, the articles were processed at the document level. For the S-HLD and C-HLD, we have processed the text at sentence and segment level (cf. Table 5.1).

For the segment level processing, we make use of (i) basic length-based chunking, (ii) topic tiling. For (i) we divide the texts into m equally sized segments based on the length of the articles. m remains fixed for all the articles. On the other hand, topic tiling makes use of topic modeling in order to segment the articles. This means for (ii), an article is considered to be a group of topics rather than a group of words. Topic Tiling uses LDA as a topic model for making n segments (cf. Section 2.4). Here, a change in topic marks the start of the new segment. n varies for each article. Since m remained static for all the articles, features could be extracted and used separately as m S-HLD. The dynamic nature of n prevented us from using features derived from each n segment as S-HLD as their number would have varied per article. Hence, we combined the features derived at the segment level into one for consistency. This combination of features was also done at the sentence level granularity since we have a different number of sentences k for every article. The method of combining is explained in detail in the Section 5.5.1.

Granularity	Technique	No. of Segments	Type
Sentence Level	Basic statistics	k	Dynamic: k changes for every article
Chunks Level	Basic statistics	m	Static: m is constant for every article
Topic Level	Topic Tiling	n	Dynamic: n changes for every article

Table 5.1: The table summarises the granularity levels and techniques used to extract them. The articles were processed at these levels for the high-level descriptors.

5.3.2 Clustering

Before extracting the high-level semantic descriptors from the articles, we group the articles into different clusters. This is done assuming that low-level descriptors failed to classify them into fake and real because their underlying distributions within each class differ from the average fake and real article. This grouping is achieved using clustering mechanisms wherein different clusters represent articles with similar underlying distributions. This clustering mechanism is used two folds i.e. (i) the clusters are used as standalone high-level descriptors to be used by the classifier. (ii) the articles within each cluster are analyzed separately using the high-level descriptors, and then their combined performance is compared to decisions made using the complete article corpus..

The text from the articles was transformed into embeddings and fed into the clustering algorithms to get the desired groups. We finally used combinations of several embeddings, algorithms, and similarity measures to get our clusters. For embeddings, we used TF-IDF, Doc2Vec, PCA, GLOVE, and LDA (cf. Section 2.1). In addition, k-Means algorithms, DBSCAN, HDBSCAN, Affinity Propagation, Spectral Clustering,

Mean Shift, and Community-based Graph Clustering (cf. Section 2.2) were used with the suitable similarity matrix (cf. Figure 5.2). They were then evaluated using the Silhouette score, Calinski Harabasz score, and Davies Bouldin score (cf. Section 2.3). The results are discussed in detail in Section 7.2. In the end, we have articles divided into various groups, which can, in turn, be used as standalone high-level descriptors or as separate groups of articles, which are then further processed separately to be able to be distinguished between fake and real.

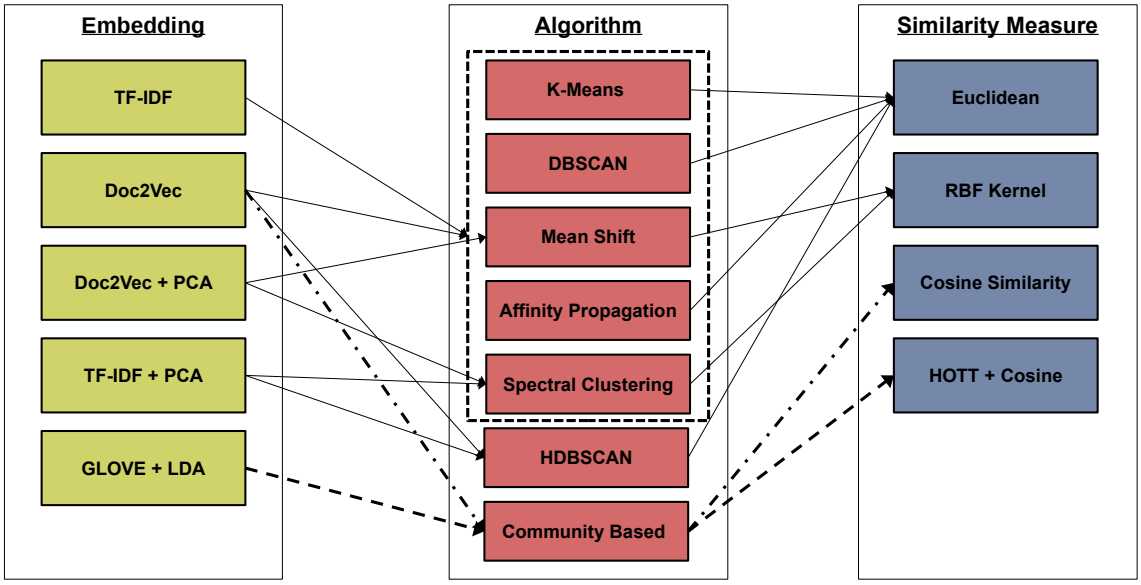


Figure 5.2: The figure depicts the combinations of embeddings, algorithms, and similarity measures used for experimentation to be able to divide the articles into separate groups to be analyzed further to classify them into Fake and Real news.

5.4 Standalone High-Level Descriptors

The primary motivation for our work is that it is a piece of very well-written fake news that poses a real threat and accounts for a smaller proportion of the current fake news being transmitted. We call it a real threat because it lacks features that can help distinguish it from legitimate news (cf. Section 3.1). To discover these hidden characteristics, we transform the text from the articles into semantically meaningful features, which we refer to in our thesis as High-Level Descriptors (HLD). As discussed above, we derive the S-HLD at different granular levels. The brief description of the explored features is as follows:

5.4.1 Attribution

The idea behind the attribution is that from a given labeled article pool (D), we find the most similar one to our query article. This pool contains articles whose labels (Real/Fake) are known to us. For our experiments, this pool is derived from the Fake News Corpus (cf. Section 4.1.3). The fakeness or realness of our query article is then attributed to the class label of its most similar article in D. The overall idea is described in Figure 5.3 wherein the demonstration is made for the granularity at the sentence level. The sentences of the articles in the labeled pool ($S_{i,j}$) are given the label

of their corresponding article. For instance, if D_j is labeled as Real, all the sentences belonging to it will also be labeled as Real ($\forall i. S_{i,j} \in D_j \Rightarrow \text{class}(S_{i,j}) = \text{real}$). The main goal of the attribution is to find the label for the query sentence ($S_{i,q}$) using the similarity measure between $S_{i,q}$ and $\forall\ S_{i,j} \in D$.

Furthermore, we wanted both contextual and non-contextual embeddings to capture the similarities. Hence, we used sentence transformers for the contextual embeddings and Doc2Vec for non-contextual ones. For the sentence transformers the pre-trained model chosen was *all-MiniLM-L6-v2* (cf. Section 2.1.4). The main motivation for choosing this model was the higher speed of execution and at par performance with other base models. This results in the 384-dimensional dense sentence embeddings.

The cosine similarity then is used to find the similarity between the sentence embedding of the query article ($E_{i,q}$) and sentence embeddings of all the articles in the labeled document pool (($\forall\ E_{i,j} \in D$). Finally, the label of $S_{i,q}$ is attributed with the class label of the most similar sentence $S_{i,j}$ within D. Furthermore, not all the sentences are labeled as fake or real. Only those sentences that surpass a certain threshold are labeled. The rest of the sentences are labeled as unknown. We have a dynamic thresholding system wherein different thresholds exist for every article, which depends upon the sentences' maximum similar cosine similarity scores. The sentences whose maximum similar scores are greater than or equal to the average of the maximum scores for the document are then labeled with their respective labels.

For the explainability part, the similarity values and attributed labels are used, but we further transform this feature into custom embeddings for the classifier to make the decision. This transformation step also exists for most of the other S-HLD. We will discuss this in detail in Section 5.5.1.

5.4.2 Topics

The basis for this feature is that certain topics are high targets for fake news (cf. Section 3.1), and hence getting to know about the topics from a given article can help in the decision-making of the classifier. For extracting the topics, two different approaches were used, which are as follows:

1. **Zero Shot Learning (ZSL):** The property of ZSL to predict the categories which were never seen by the learner (cf. Section 2.6.3) is exploited to find the topics within the article. The topics were extracted at all levels of granularity, namely sentence, chunk, segment, and document levels. For our thesis, we fixed these categories to war, government, politics, education, health, environment, economy, business, entertainment, and sports. We decided on these categories based on the most popular topics in the news articles (cf. Section 3.1).

   For our experiments, we have used the *facebook/Bart-large-mnli* model based on the method proposed by Yin et al. [2019], which shows that ZSL works very well with the BART model. The ZSL classifier then outputs the normalized score for all the categories, which is used as our classifier's feature. The document level topic scores are used for the feature, but for the other granularity levels, we label the sentences/segments/chunks with the category having the maximum score. These labels are combined with other features like attribution and entailment, which are discussed in detail in Section 5.5.2.

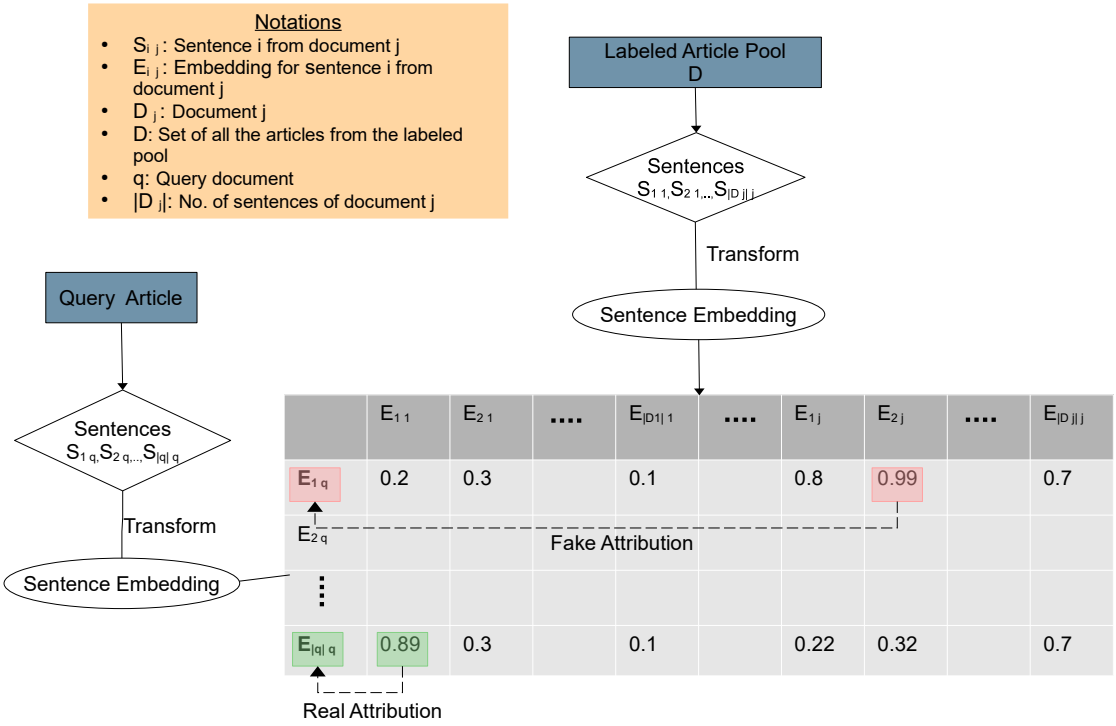


Figure 5.3: The figure shows the process of using attribution as a feature. The documents are split into sentences, and then the text is transformed into embeddings. The similarity is computed between the query sentence and all the sentences of the labeled document pool. The label (Fake/Real) of the query sentence is then attributed to the label of its most similar sentence from the pool.

2. **Topics using MPQA Dataset:** The MPQA dataset (cf. Section 4.1.5), along with the documents, contains additional meta information like the source, country, and the topic of the article. The meta-information about the topics and the documents provided are used to train a classifier. Finally, the trained classifier is used to predict the topics of our articles. However, for the MPQA dataset, there were multiple topics for a single document (multiple labels) and multiple classes for the entire dataset. Hence, we had to train a multi-output classifier on the same dataset that considers multiple classes and labels. For representing our text, we encoded the same using sentence transformers. The model underlying the same was *Microsoft/deberta-v2-xlarge-mnli*. This model was chosen because of its outstanding performance in NLU tasks on the benchmark datasets[34]. Also, we compared the results using the *all-MiniLM-L6-v2* model, and DeBERTa performed better than it in training accuracy. Also, we used TF-IDF embeddings for the baseline comparisons.

   Here, the document-level prediction is straightforward, wherein the normalized output of the prediction function of the multioutput classifier was directly used as a feature. For other granular levels, we wanted one label per level. Hence, the label predicted with maximum probability was then used as a topic for the sentence or segment. These labels/topics for each sentence/segment, just like ZSL, were then used in combination with other features (cf. Section 5.5.2)

[34] https://huggingface.co/microsoft/deberta-v2-xlarge-mnli

5.4.3 Personality

The articles also tend to depict different personality traits of the writer, as seen in Li and Chignell [2010]. For this reason, we use the MBTI dataset (cf. Section 4.1.5) in order to predict the language style concerning the personality, which might vary for the fake and real articles. We have transformed the labels of the MBTI dataset into a multi-label scenario with labels being extrovert, sensing, feeling, and perceiving. The other set of labels, introversion, intuition, thinking, and judging, is complementary to the chosen labels. This means the presence of extroversion implies the absence of introversion. An example of this transformation is shown in Table 5.2. The classifier is then trained in the multi-label setting using the attributes as TF-IDF embeddings on documents of the MBTI dataset and transformed labels as our prediction personality types. The personality feature is extracted only at the document level. The prediction from the multi-label classifier on our article is directly used as the feature to decide between fake and real articles.

	Multi Label Transformation			
MBTI Type	Extrovert	Sensing	Feeling	Perceiving
INFJ	0	0	1	0
ENTP	1	0	0	1

Table 5.2: The table demonstrates the transformation of the personality type of the MBTI dataset into a multi-label setting. Here, the presence of the personality trait is encoded as a 1 and absence as 0. For INFJ, the presence of Introversion, Intuition, and Judging implied the absence of Extrovert, Sensing, and Perceiving. Hence, these labels were encoded as 0.

5.4.4 Parts of Speech Structure using Markov Models

The types of parts of speech (POS) used in an article are known to form the basis of the writing structure of an article (cf. Munro [2006]). This led us to use this intrinsic structure as a decision criterion, separating real and fake articles. For the training purposes of this feature, our training articles are separated based on their label (Real/Fake). The articles within each category are considered one extensive article and are divided into sentences. For words within each sentence, we extract their POS tags and append a start and end tag to mark the beginning and end of the sentence, respectively.The bigrams of the POS tags are then used to train the Markov models (cf. Gagniuc [2017]) for real and fake categories (cf. Figure 5.4).

When the query article is observed, it goes through the preprocessing steps involving conversion into sentences, the addition of start and end tokens to the sentences, POS tag extractions for words in the sentence, and finally, extracting the POS bigrams of the sentences. The probability of occurrence of POS bigrams is then fetched from the trained real and fake Markov models. The Markov model having the greater probability, in turn, governs the label of the sentence. Finally, the article's label is given according to the mode of real and fake sentences.

We have demonstrated the same with an example in Figure 5.5 wherein we have seen how the Markov model makes predictions for the label of the sample sentence *Earth*

is flat. For a query article, all sentences are labeled in the same way, and the label of the majority of the sentences is then used as the query article's label.

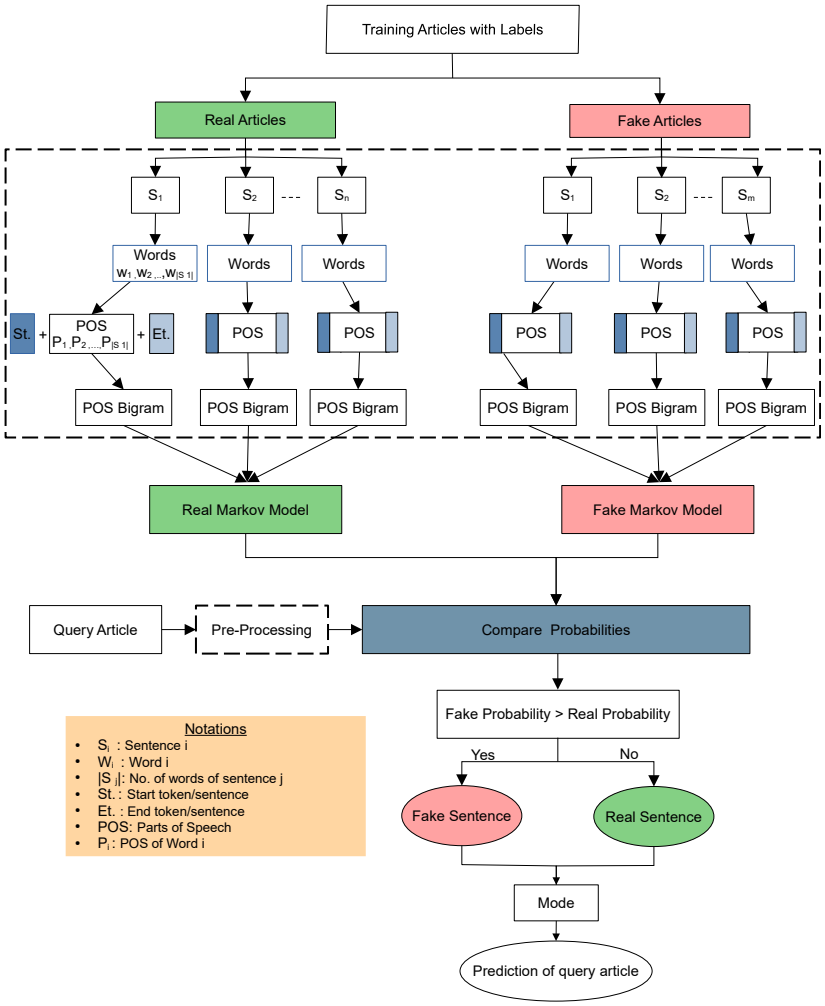


Figure 5.4: The figure depicts the usage of POS and the Markov models to determine the underline structure of the fake and real articles. The training phase involves creating Markov models for the real and fake categories using the POS structure of sentences in the training data. In the testing phase, we use the trained Markov model to predict the POS structure of the sentences in the query article.

5.4.5 Sentiment, Tone and Entailment Analysis

The articles are written to engage the audience and influence the readers to believe the article's contents. Sentiments potrayed in the text play a major role in audience targetting (Aslam et al. [2020]). Hence, we wanted to analyze if sentiments as part of textual features can help distinguish between fake and real articles. We use the pipeline for *sentiment analysis*[35] made available by the hugging face transformers module. The model used for our analysis is *distilbert-base-uncased-finetuned-sst-2-english* (cf. Section 2.6.2). The reason for choosing the same was its self-supervised

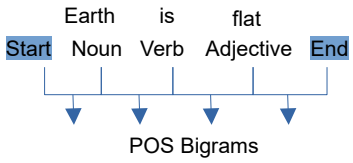
[35] https://huggingface.co/docs/transformers/main_classes/pipelines

Trained Real Markov Model - RMM

POS Bigram ($P_{|S\,j|-1}$, $P_{|S\,j|}$)	Count of ($P_{|S\,j|-1}$, $P_{|S\,j|}$) (x)	Bigrams starting with $P_{|S\,j|-1}$ (y)	Probability (x/y)
(Start, Noun)	300	900	1/3
(Noun, Verb)	200	600	1/3
(Verb, Adjective)	300	1200	1/4
(Adjective, End)	200	800	1/4

Trained Fake Markov Model - FMM

POS Bigram ($P_{|S\,j|-1}$, $P_{|S\,j|}$)	Count of ($P_{|S\,j|-1}$, $P_{|S\,j|}$) (x)	Bigrams starting with $P_{|S\,j|-1}$ (y)	Probability (x/y)
(Start, Noun)	200	400	1/2
(Noun, Verb)	200	800	1/4
(Verb, Adjective)	400	1200	1/3
(Adjective, End)	100	200	1/2



Prob. according to RMM: $\frac{1}{3} \times \frac{1}{3} \times \frac{1}{4} \times \frac{1}{4} = 0.0069$

Prob. according to FMM: $\frac{1}{2} \times \frac{1}{4} \times \frac{1}{3} \times \frac{1}{2} = 0.02$

P(FMM) > P(RMM) → Sentence is labeled as Fake

Figure 5.5: The figure demonstrates, with an example, the work described in Figure 5.4. Here the sample sentence *Earth is flat* is labeled fake since the probability of the bigrams from the trained fake Markov model is greater than that of the real Markov model.

training and ability to produce faster results. We have used this pipeline at the sentence level, wherein it outputs the label for the sentence as positive or negative along with its score. This score, along with the label, is then used as a feature. To get the feature at the document level, we take the average of the scores of each sentence.

Additionally, to the sentiments, we wanted to see if the article's tone could help find the semantic differences between the fake and real articles. The sentiments tend to give a notion of general feelings wherein tone is used to represent the mood formally. For the tone analysis, we used the tool by IBM, *Tone Analyzer*[36]. The analyzer outputs the tones and the score for the tone for both documents and sentences, and we used both the results for the feature extraction. The tones marked by the tool are *Analytical*, *Anger*, *Confident*, *Fear*, *Joy*, *Sadness*, and *Tentative*. After we have marked the tones for the articles and the sentences, we need to transform the feature into the embeddings, which is talked about in detail in Section 5.5.1

Furthermore, real articles are known to have sentences that are backed up by facts or related content. This means that succeeding sentences in the real articles should entail the preceding sentences backing up the statement. In the fake articles, it is assumed that the number of contradictory statements will overpower the number of entailing statements (cf. Unkelbach et al. [2019]). To use entailment as a feature, we created sentence pairs for our articles and saw if the following sentence in a pair entails, contradicts, or is neutral to the preceding sentence. In order to do so, the hugging face pipeline of *text-classification*[37] is used. The model used for the same is *roberta-large-mnli* since it optimizes the BERT's pre-training approach, resulting in faster and better performance (cf. Section 2.6.1). The pipeline outputs the labels ENTAILMENT, CONTRADICTION, and NEUTRAL along with their scores, which

[36] https://cloud.ibm.com/apidocs/tone-analyzer
[37] https://huggingface.co/tasks/text-classification

indicate the weightage of the label output. We get the labels and score for each sentence pair. To get the labels at the document level, we take the average of the scores of the sentence pairs to get a single embedding. We discuss this in detail in Section 5.5.1.

5.4.6 Words Net's Hypernym - Hyponym Structure

The word usage in articles also has an impact on biasing the reader. Also, the articles that do not have enough claims to support their content tend to use general words to indicate something (hypernym). On the contrary, articles with strong backing statements tend to use specific words (hyponyms) (cf. Kapusta et al. [2020]). We wanted to check this hypothesis and extract the information about the words in terms of whether they are hypernyms, hyponyms, or words between both. This information about the words is extracted using the hierarchies in the WordNet (cf. Section 2.7).

This analysis is done at the segment level, using basic length-based chunking. We have the same number of chunks m for every article (cf. Table 5.1). This ensures that all the documents are comparable at the chunk level. We analyze the words within each chunk and categorize them as *General*, *Specific*, or *In-Between* words. Also, to ensure we capture the context of the words and map them to the correct categories, we use the lesk algorithm to get the correct mapping for the hypernyms and hyponyms. We then count the occurrences of each category within the chunk and use the normalized counts as the embedding for each chunk, the general process for which is described in Figure 5.8. We, in the end, have the embedding of length $m \cdot 3$ for each document, wherein three stands for one category General, Specific, or In-between.

5.4.7 Custom Embedding using the categories of Parts of Speech

Building on the usage of the types of words, we used the parts of speech of the words to see what categories the words belong to. For this feature, we used only nouns, verbs, and adjectives. The categories are also used to aid in explainability since it reduces the dimensionality from word level to a category level which is easier to comprehend. For our analysis, nouns and verbs go through one kind of feature engineering (cf. Figure 5.6), and for adjectives, another kind of feature engineering (cf. Figure 5.7).

We used categories predefined by the WordNet (cf. Table 5.3) for nouns and verbs[38]. For nouns, we divided the words into 26 categories and verbs into 15. For extraction of POS from the words, we observe the words at the sentence level. The reason is that POS might be different in a different context. For example, the word *light* is NOUN in the sentence, *The LIGHT was so bright that it hurt my eyes* whereas it is VERB in the sentence, *I asked him to LIGHT the way with the help of a torch.* Once the nouns and verbs are extracted from sentences, they are mapped into their respective categories. After that, we count the categories within each sentence and transform them into embeddings at the document level (cf. Figure 5.8).

Since there are no predefined categories available for adjectives, we grouped the inbuilt adjective categories into eight main groups via k-Means clustering (cf. Figure 5.7). The

[38] https://wordnet.princeton.edu/documentation/lexnames5wn

satellite adjectives were not considered for clustering; instead, their head adjectives were used as we wanted to have fewer categories for explainability purposes. For the k-Means, the words were transformed into their corresponding word embeddings using GLOVE. Then, when the query article comes, we extract from the sentences the satellite adjectives and use the trained k-Means model to predict the category in which the adjectives fall. After that, the same steps as that of noun-verb extraction were used where we counted the categories within each sentence and transformed them into embedding at the document level. Every query article has three custom embeddings w.r.t. to the noun, verb, and adjective categories. They were explainable low-dimensional embeddings.

Id	Noun Categories	Verb Categories
1	GENERAL	BODY
2	ACT	CHANGE
3	ANIMAL	COGNITION
4	ARTIFACT	COMMUNICATION
5	ATTRIBUTE	COMPETITION
6	BODY	CONSUMPTION
7	COGNITION	CONTACT
8	COMMUNICATION	CREATION
9	EVENT	EMOTION
10	FEELING	MOTION
11	FOOD	PERCEPTION
12	GROUP	POSSESSION
13	LOCATION	SOCIAL
14	MOTIVE	STATIVE
15	OBJECT	WEATHER
16	PERSON	-
17	PHENOMENON	-
18	PLANT	-
19	POSSESSION	-
20	PROCESS	-
21	QUANTITY	-
22	RELATION	-
23	SHAPE	-
24	STATE	-
25	SUBSTANCE	-
26	TIME	-

Table 5.3: The table summarises 26 Noun and 15 Verb categories used to create the low-dimensional explainable custom embeddings. The categories are taken from WordNet's Database.

5.5 Further Processing of High-Level Descriptors

We use the high-level descriptors from the previous section to derive additional descriptors for our pipeline. These are composed into transformed variants, i.e., those where individual sentences and segments are rolled up to a single feature vector, and combined versions, i.e., those descriptors where at least two high-level descriptors are combined into one new descriptor.

5.5.1 Transformed Standalone High-Level Descriptors

We saw in Section 5.4 that we had highlighted the need for transforming features into the embedding to be used for the classifier. This step helped us to use the features at different levels of granularity uniformly across all the segments. This was important because the number of sentences and segments varied across various articles (cf. Table 5.1). To have a feature at a consistent level, we had to aggregate the same at the article level. The features at the article level are also transformed into embedding so that instead of dealing with categories as in topics, tones, and sentiments, we use their occurrences as numerical attributes to be, in turn, used by the classifier. The process of transformation is described in Figure 5.8.

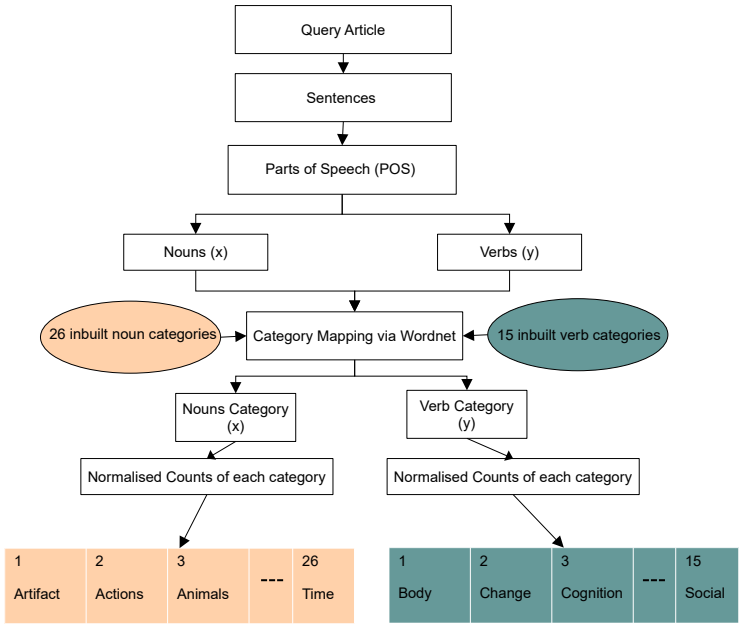


Figure 5.6: The figure demonstrates the process of extracting the Noun and Verb embeddings from sentences. We make use of the parts of speech of words inside the sentence and then map them into the categories described in Table 5.3.

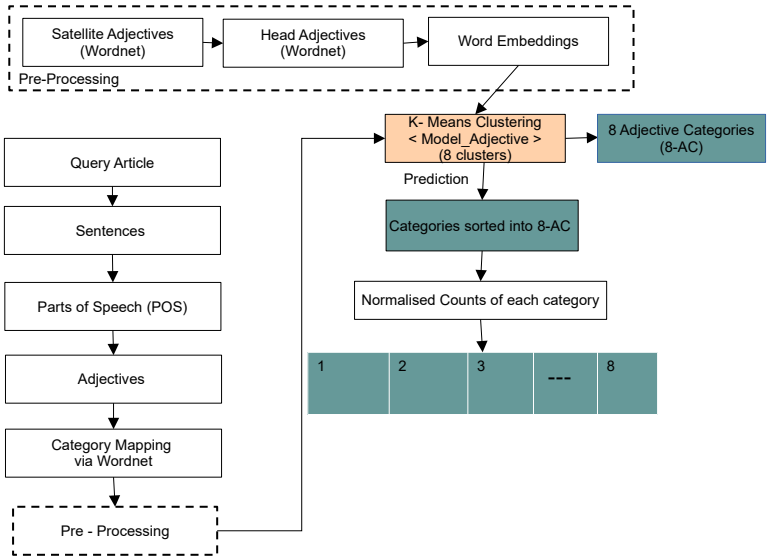


Figure 5.7: The figure describes the process of extracting the Adjective embeddings from the sentences. Due to the absence of predefined adjective categories, we made our eight meta categories by clustering the head adjectives of the satellite adjectives.

5.5.2 Combined High-Level Descriptors:

Instead of using stand-alone features, we wanted to see if they could help with decisions when used together. For this, we combine the features of attribution and entailment with other features. The list of combinations is described in Table 5.4.

Combining the features involves transforming the unigram categories of attribution, topics, and sentiments into bigrams. First, the entailment labels are used without

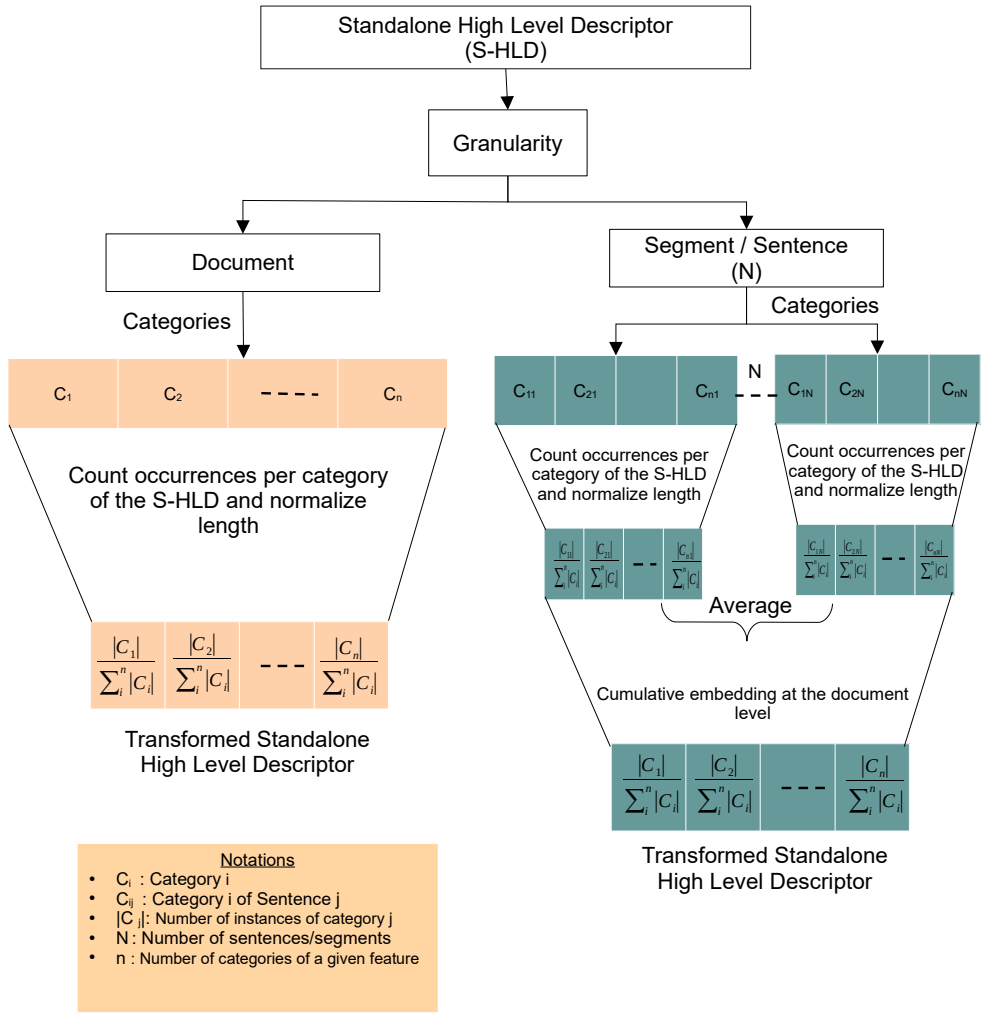


Figure 5.8: The figure demonstrates the process of combining features at different levels of granularity into one single embedding. This is done to have a uniform feature length for the classifier across all the articles.

S-HLD$_1$	S-HLD$_2$
Attribution	Entailment
Attribution	Topics
Entailment	Topics
Attribution	Sentiments
Entailment	Sentiments
Attribution	Tones
Entailment	Tones

Table 5.4: The table shows the combination of features that were used together as part of the thesis as described under Section 5.5.2

transformation because they implicitly use bigram properties by using sentence pairs as an input (cf. Section 5.4.5). We then combine the set of features shown in Table 5.4 using the bigrams. We then count occurrences of combined categories at different

granularities and use the normalized counts as the final feature embedding. The example for combining Attribution and Entailment is shown in Figure 5.9.

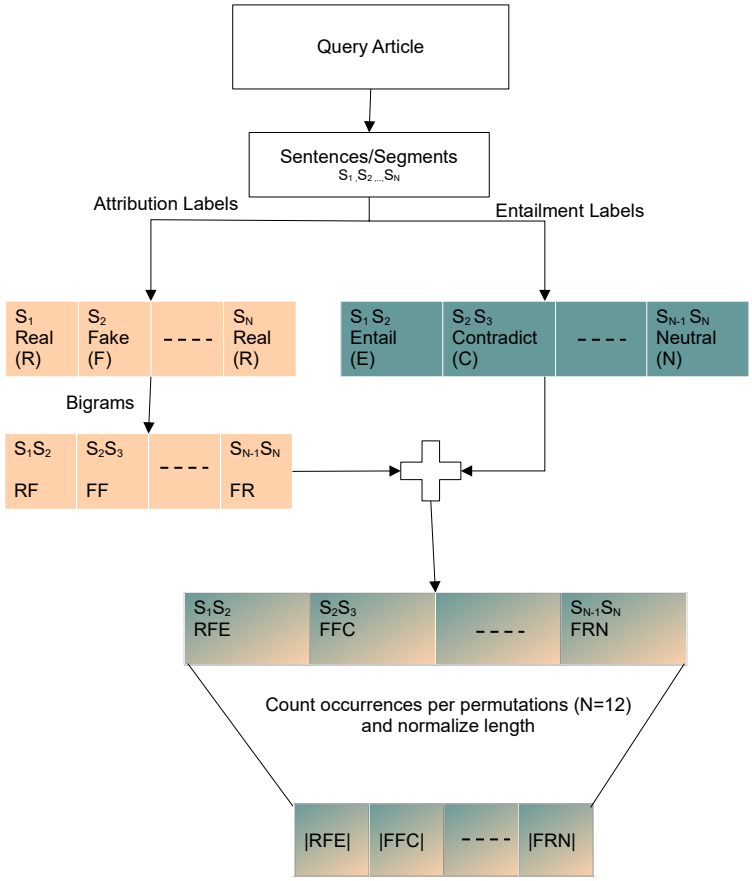


Figure 5.9: The figure demonstrates the process of combining features Attribution and Entailment to be used in a combined form for the classifier. The combination shown is true for both sentence and segment level granularity. This process can generally directly combine any set of features for the classifier's use.

5.6 Explainability

To incorporate the explainability into our system, we used intrinsic methods via the features extracted for the classifier and extrinsic methods via quantitative and qualitative practices. For instance, for the custom embedding (cf. Section 5.4.7) used for the noun, verb, and adjective categories, we know precisely the meaning of numbers in the embedding compared to the classic word/sentence embeddings. In another example, wherein we use parts of speech structure (cf. Section 5.4.4) to unravel the article's realness or fakeness is also a very explainable feature in terms of structural analysis. Similarly, other low and high-level descriptors are implicitly designed to make explainable decisions. These explainability factors cater to the steps before being put into a classifier. However, we also wanted to focus on explainability after the decisions made by the classifier. That is where the quantitative and qualitative methods dive in.

5.6.1 Quantitative

The quantitative methods include the usage of partial dependence plots, feature importance scores, and SHAP value. In addition, we made use of the interactive tool *Explainer Dashboard*[39] which helps us visualize the quantitative measures all at once. The results for the same are discussed in detail in Section 8.3.

1. Feature Importance: As seen in Section 2.5.2, feature importance is a widely used method for finding the attributes that contribute the most towards the classifier's prediction. This, in turn, aids in explainability by enabling the user to understand the essential features for a particular prediction and build trust with the system. Furthermore, knowing the essential features can let a user scrutinize a particular part of the article to validate the facts. For example: if it shows that an article is fake because of its attribution to fake sources, one can cross-check this information by going through the alleged sources to make sure if the prediction makes sense or not.
2. Partial Dependence Plots (PDP): PDP aims to form a link between the target label (fake/real) and the attributes used to predict the target variable (Low-Level/High-Level Descriptors). PDP plots show how the prediction change by changing the value of a single feature (cf. Goldstein et al. [2015]). Since there are only certain features that affect the final prediction, we experiment with the features with the maximum Feature Importance score. The best part about the PDP plots is that they are model agnostic, i.e., they can be used with any machine learning algorithm. Furthermore, it is also a global method, i.e., all the observations in the dataset are considered while presenting the output.
3. SHapley Additive exPlanations (SHAP): SHAP values are the state-of-the-art technique also used to figure out the effect of each attribute on the classifier's prediction. The concept has its root in the gaming theory and considers the reproduction of the classifier's predictions as a *game* and attributes used for the prediction as the *players*. Just like the Shapley values in the game theory, SHAP aims to quantify the contribution of all the attributes to the classifier's prediction (cf. Lundberg and Lee [2017]). For our thesis, we use the contribution plots made by the SHAP values, wherein we show the contribution of all the attributes towards its prediction for each observation.

5.6.2 Qualitative

We merged the features and the classifier's decision for the qualitative aspect. For instance, the attribution feature was used to enhance qualitative explainability. The sentences were highlighted according to their attribution labels (cf. Figure 5.3). Also, the similarity scores were used as weights to give different gradients according to the similarity value. The attributed source with its content is shown additionally. This was also incorporated into the user interface, which can be seen in detail in Figure 6.2. A small example of the same is demonstrated for a news article by Indian

[39] https://explainerdashboard.readthedocs.io/en/latest/

Express titled *Monkeypox virus could become entrenched as new STD in the US*[40] in Figure 5.10.

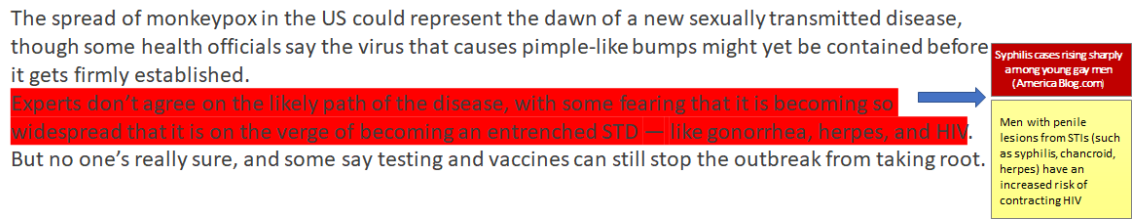
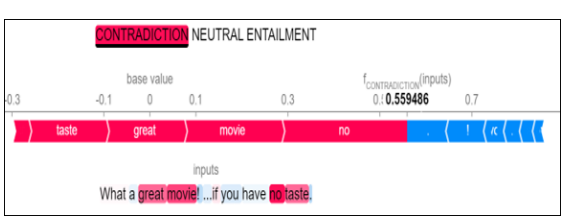
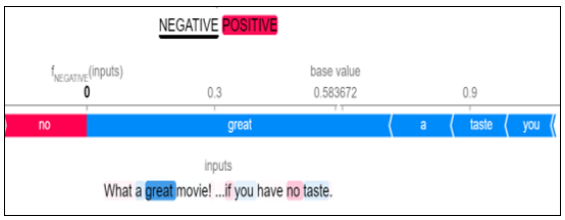


Figure 5.10: The figure demonstrates how attribution as a feature can be used to aid explainability. Here the sentence is labeled and hence colored according to the category (real or fake) attributed to it. The red color in the given example means that the sentence "Experts don't agree on the likely [...]" has been attributed to the sentence from the fake sources, which in this case is the news article titled "Syphilis cases rising sharply among young gay men" from the known fake source *America Blog*. Furthermore, the darkness/lightness of the hue indicates the similarity with the attributed source. More is the similarity; darker is the shade of the labeled sentence.

Similar explanations were made using the features Entailment and Sentiment using the SHAP plots. A small example of the sentence "What a great movie!...if you have no taste." is demonstrated in the Figure 5.11. Here we could see which part of the sentence is responsible for the statement being contradictory or having a positive sentiment. This enables system transparency, giving the end user a chance to validate the same. Not only does this system help the end user, but also the developer. For example, the sentence is labeled as having a positive emotion, whereas it is clear to the reader that it is the sarcastic sentence that is not captured. This can point us toward what other techniques can be experimented with to overcome the limitations.



(a) The subfigure shows how the words *great movie* followed by *no taste* contribute towards contradiction in the given sentence.

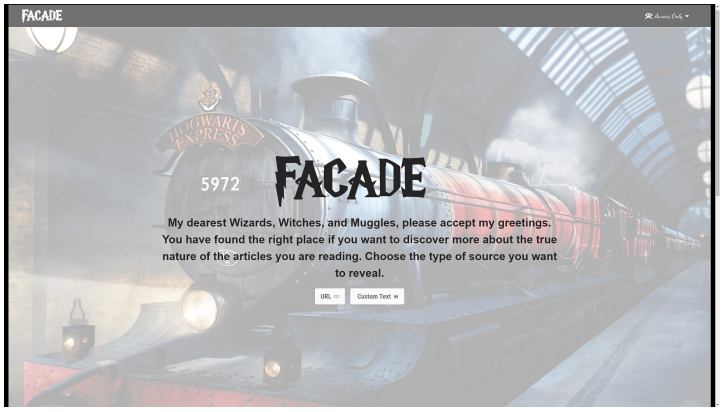


(b) The subfigure shows the influence of *great* and *taste* in making the sentence positive. The word *no* brings in the negative sentiment but is overpowered by the positive value.

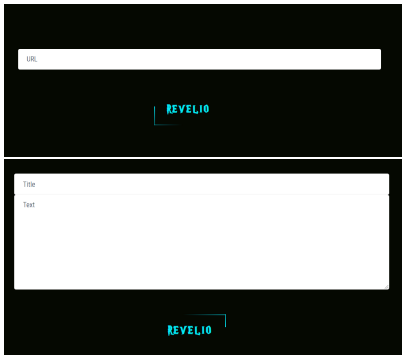
Figure 5.11: The figure shows why the sentence, "What a great movie!...if you have no taste", has been labeled to have a higher contradiction and positive score by the Entailment and Sentiment pipelines making use of transformers. The results were in sync with the words highlighted. However, the models could not capture the sarcasm in the sentence, which would have made the sentence more likely to have negative emotions.

[40] https://indianexpress.com/article/world/monkeypox-virus-could-become-entrenched-as-new-std-us-8046596/

6. User Interface



(a) The subfigure shows the front page of the UI.



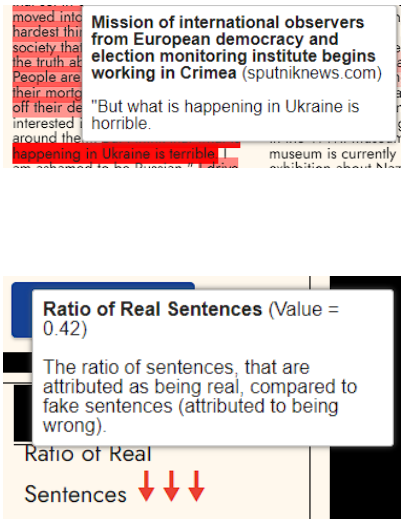
(b) The subfigure shows the URL input on top and the custom text input on the bottom.

Figure 6.1: The figure shows the initial view for our UI for FACADE. (a) shows the welcome page, whereas (b) depicts the two possible input options, with the URL input on top and the input for custom text on the bottom.

To show how a potential system of fake news detection could look in practice and provide some examples for explainability, we designed a prototypic user interface. Of course, the user interface is not in a production-ready state since it still does not cover all relevant options and is comparatively slow, but it can give some insights into what is possible. When the user opens the website, they are greeted with a welcome page and may choose to either input a URL or a custom text for evaluation. This front page and the forms for those two options can be seen in figure 6.1. Additionally, on the top right, the advanced user will find two links: the first one will point towards the GitHub of the Fake News Corpus, and the second one towards a version of the ExplainerDashboard, that was set up on our model along with the 4076 test instances. It is there to provide a deeper insight into the inner working of our model, which will be discussed in a later section 8.3. After entering either the URL or custom text

and clicking on the submit button called *Revelio*, messages are sent to the UI to show the current state of the calculations, e.g., that pipeline one is currently running or that attributions are being calculated. These messages can be easily adjusted to fit the corresponding type of user and use case. For example, if the news article could be confidently classified just using the first pipeline, the user is directly sent to the result page. Otherwise, they are being presented with the choice to go on with the following pipeline (in a slower or faster variant, i.e., with or without zero-shot learning) or stop at the current (in confident) state. Either way, in the end, they would land on the result page eventually.

(a) The subfigure shows the result page of the UI.



(b) The subfigure shows two potential tooltips: Above is the tooltip, which is displayed when hovering over an attributed sentence. When hovering over the features, the lower tooltip is shown, which led to the decision for this article.

Figure 6.2: The figure shows the initial view for our UI for FACADE. (a) shows the welcome page, whereas (b) depicts the two possible input options, with the URL input on top and the input for custom text on the bottom.

The result page can be seen in figure 6.2a. At first, the user is presented with only the feature contributions on the right-hand side, whereas the arrows indicate how strong the influence of that feature is (by the number of arrows, whereas three arrows are the maximum) as well as whether it pushed it towards real (green, up arrow) or fake (red, down arrow). If the user clicks on either the *Show Real* or *Show Fake* Button, the sentences attributed to either of the two classes are shown. The button *Show Real* was already clicked in the figure. Therefore, the red highlighted sentences are the ones that are being attributed as being fake. Since the user might need an

explanation on why a particular sentence is being attributed as fake, they can hover over that sentence and see the attribution. This can be seen in the figure 6.2b on top. Here the sentence is “But I think that what is happening in Ukraine is terrible”, which is attributed to being fake since there is a fake news article which has a very similar sentence stating “But what is happening in Ukraine is horrible”. Additionally, there are tooltips when the user hovers over the feature contributions on the right-hand side. For example, it can be seen in the figure 6.2b below and explains what that feature means (here, the attribution feature is explained) and the actual value of this feature. Of course, the raw value does not help that user yet. For example, it might be explained later in the stages by providing a graph on how to place that value or replace it with a more straightforward categorical value like low, medium, and high.

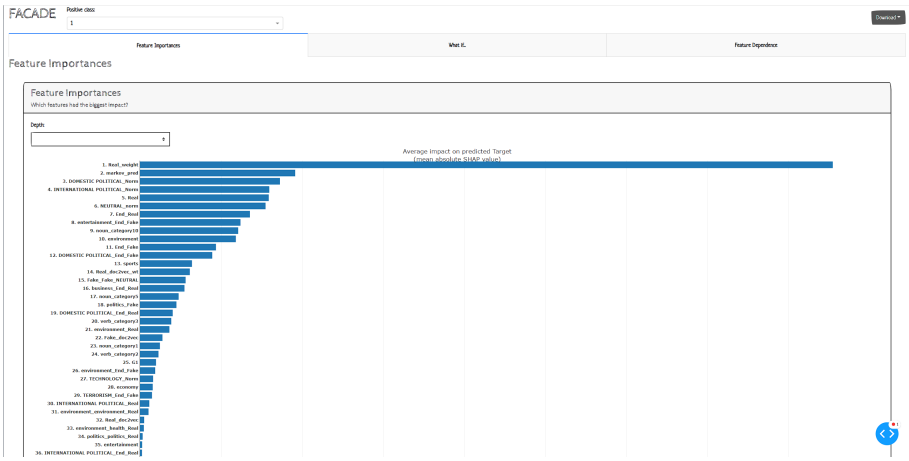


Figure 6.3: The figure shows the ExplainerDashboard, when it is being used for only a single instance, which the user provided through the UI. It is shown upon clicking the *More Explanation* button.

Last but not least, the user may click on the *More Explanation* button, which will lead to an ExplainerDashboard specifically only for that particular instance provided. It will provide more insight, as was explained in section 5.6.2 and will be additionally discussed later in section 8.3. A particular instance can be seen in figure 6.3, which was generated on the same (wrongly classified) instance as from the previous figures. Here it is shown which other features influenced the decision and how high the difference is. I.e., the influence of the few real sentences being attributed in that text is such a significant decision factor that it overpowers all other criteria. The user might also click on the *What if...* page to change the features and therefore see what could be changed in the article so the system can understand that it is a real article. In this case, however, it is clear that the corpus is probably biased towards articles about Russia and Ukraine being mostly fake articles and therefore having many fake sources to attribute the sentences to.

7. Evaluation

7.1 Experimental Setup and Materials

In order to implement the concepts defined in Section 5, specific tools were used to achieve the desired results. Although we aimed to minimize pre-trained tools to increase transparency and explainability, some features used existing libraries. In this section, we will briefly discuss the libraries and the selection of the parameters for the same. Also, we discuss the results we achieved at each phase of our pipeline.

7.1.1 Low-Level Descriptors: Tools, Setup, Results, and Evaluation

7.1.1.1 External Tools and Setup

The first step in generating the Low-Level Descriptors (LLD) was text preprocessing, which included tokenization, lemmatization, and text-specific cleaning. In order to do so, we used the functionalities of the standard NLTK[41] library. Furthermore, in order to figure out the grammatical mistakes, we used the *language tool python*[42], which helps in figuring out the possible grammatical mistakes in the text in the sentence level granularity. For TF-IDF as a feature, we made use of the functionality of a widely used library named *scikit-learn*[43]. After deriving all the features as described in Section 5.2, they were used by the classifier in the first phase of the pipeline (*Classifier-1*) to see how they help distinguish between the fake and real articles. In our thesis, we experimented with various algorithms such as Random Forest, XGBoost, Gradient Boosting, and Decision Trees, as discussed in Section 2.5. We selected the final model based on accuracy as a performance criterion. *Classifier-1* was additionally trained on only TF-IDF representation of text as features for well-established baseline comparisons.

[41] https://www.nltk.org/
[42] https://pypi.org/project/language-tool-python/
[43] https://scikit-learn.org/stable/

We divided our dataset into train, validation, and test split for learning. The extra validation step was introduced mainly for selecting the filtering thresholds. For the Fake News Corpus (c.f. Section 4.1.3), we divided the corpus such that we had 34744, 8788, and 8766 news articles for training, validation, and testing, respectively. We only state the Fake News Corpus here because we chose not to use the Kaggle and the ISOT dataset because of biased and easy-to-detect content. The experiments to briefly discuss the same are shown in Section 7.1.2.

7.1.1.2 Results and Evaluation

After conducting the experiments with the classifier algorithms mentioned above, *Random Forest* performed the best with TF-IDF experimentations, and XGBoost overpowered for the experimentations involving LLD. The baseline of TF-IDF gave the test accuracy of 84.50% by using 50,000 words as features. Because 50,000 features result in a very sparse embedding and a loss of explainability, we want to reduce the number of words based on their importance to the classifier's decision. The chi-square test (cf. Meesad et al. [2011]) was used to find the top 50 TF-IDF scores influencing the classifier's decision.

The features with the highest top 50 chi-square values were then selected. The testing accuracy for the reduced set of TF-IDF features was reduced to 69.61%. The 27 LLD stand-alone gave an accuracy of 77.36%. We then integrated the LLD with the reduced TF-IDF features to get the test accuracy of 80.33%. However, this accuracy was low compared to the TF-IDF features; the number of features was reduced significantly to 77, which aids in explainability. The results from the first phase of the pipeline using *Classifier-1* are compared in Table 7.1.

Features	Number of Features	Model	Train Accuracy (34744)	Validation Accuracy (8788)	Test Accuracy (8776)
TF-IDF	50000	Random Forest	97.2%	85.02%	84.50%
TF-IDF reduced	50	Random Forest	77.5%	70.16%	69.61%
LLD	27	XG-Boost	89.38%	76.99%	77.36%
LLD + TF-IDF reduced	77	XG-Boost	90.59%	80.81%	80.33%

Table 7.1: For the first phase of the pipeline, we use TF-IDF features and its variants as a baseline for our proposed approach (LLD). For training, we had in place 34744 news articles, of which 56.2% were real and 43.8% were fake. The validation and testing were done on 8788 and 8776 news articles, respectively, with both sets sharing the real and fake proportional ratios as seen in the training set.

7.1.2 Dataset Selection Criterion

As seen in Section 4, we had 3 data sets to experiment with, namely, the Kaggle dataset, the ISOT dataset, and the open source Fake News Corpus. To select the dataset for our primary thesis analysis, we used the LLD as the manner of elimination. However, if the LLD could easily distinguish between real and fake news, the data would be biased because it is created using fixed samples of real and fake news, which can be easily detected. Therefore, using such datasets defeats our purpose of solving the actual menace of detecting very well-crafted fake news.

With the Kaggle and ISOT datasets with only LLD, we achieved an accuracy of approximately 94% compared to only 77.36% on the fake news corpus. Furthermore,

adding the reduced TF-IDF set to our LLD increased the accuracy of Kaggle and ISOT datasets to 96% compared to only 80.33% for the Fake News Corpus. The results were in sync with the EDA done in Section 4.2 wherein we saw that Kaggle and ISOT talked about fixed domains and showed differences w.r.t. the word and length statistics between fake and real news. Related work done on the ISOT data set also showed, that high accuracies can be easily achieved, e.g. 99% for a CNN-RNN approach by Nasir et al. [2021].

The reason LLD could not help differentiate the news articles in the Fake News Corpus proved its unbiased nature. This is also because for this particular dataset, the news articles were curated from approximately 9,400,000 domains and were not stringent towards a piece of news from a particular topic. Additionally, since we added the biased articles to the pool of fake articles for our analysis, the LLD would not be able to find the common distinguishing factor between real and fake since it would also include biased news. This needs the power of semantics to capture the internal structure and hence is well suited for our proposed approach. All the further experiments were hence made using the Fake News Corpus.

7.1.3 Filtered Thresholds

As discussed in Section 5.1.4, the filtering mechanism is in place to ensure we are confident in our predictions, for which we calculate t_1 and t_2 for getting confidently fake and confidently real articles. The results of the baseline and dynamic threshold methods are discussed as follows:

1. **Baseline:** In our experiments t_1 was 0.2 and t_2 was 0.79. Herein, every article with a predicted probability below 0.2 was labeled as a confident fake, and every article with a prediction probability above 0.79 was labeled as confident real. All articles with prediction probabilities between 0.2 and 0.79 were labeled borderline cases. The accuracy of the confident fake and real articles was 92.11%.
2. **Dynamic Threshold:**
   (a) In our experiments, since we have a balanced dataset, the minimum accuracy we can expect is 50%. Also, as we see in Table 7.1, the accuracy without any thresholds using LLD in combination with reduced TF-IDF features is approximately 80%. The thresholding should hence be able to help achieve the $A_{min}^{conf} > 80\%$. This jump in accuracy is more critical and happens relatively slowly compared to a change in the number of borderline cases. Hence we, for our experiments, give more importance to w_0 and the fix w_1 as 1 making use of the Eq. 5.1.
   (b) The experiments for α and β were then done, using different combinations of thresholds between 0 and 1 at an interval of 0.001, i.e., 1000×1000 combinations of the thresholds. The combinations that did not make sense meaning wherein $t_2 < t_1$ were discarded. For finding the w_0, we relied on the graph visualizations to select the best parameter. We plotted α and β concerning different w_0 values ranging between 1 to 10 and then chose w_0 wherein we had the harmony between α and β, i.e., the accuracy rise should not be at the cost of many articles being left out as borderline cases. In doing the experiments, w_0 was chosen as 4.8.

(c) Finally, t_1 and t_2 were selected then based on the maximum value of $O(t)$ as 0.183 and 0.841 respectively. This resulted in the A_{min}^{conf} of 93.08% on the test set over 4700 articles. The remaining 4076 were then marked as borderline cases and sent to the pipeline's second phase for further analysis (c.f. Table 7.2).

	Before Threshold Articles	Before Threshold Accuracy	After Threshold Articles	After Threshold Accuracy	Border Line Cases
Validation Set	8788	80.81%	4765	92.81%	4023
Test Set	8776	80.33%	4700	93.08%	4076

Table 7.2: We wanted to test out our filtering mechanism for the best thresholds on the classifier making use of LLD + reduced TF-IDF set as features. We used the validation set of 8788 articles for the main experimentation. The selected thresholds were then used for testing the accuracy on the test set of the confidently predicted articles. With the selected thresholds we were able to confidently predict 4700 articles with 93.08%.

7.2 Clustering Quality

We saw in Section 5.3.2 that we made the groupings of the articles labeled as borderline cases in the first phase of the pipeline. These groupings were made to see the patterns within the articles, which were not quickly detectable using the LLD. For finding these groupings, we used various clustering algorithms, using different combinations of text embeddings and similarity measures, as seen in Figure 5.2. For evaluating each clustering algorithm, we used intrinsic evaluation algorithms that do not require the ground truth for the evaluation, such as the Silhouette Score, Calinski Harabasz Score, and Davies Bouldin Score (c.f. Section 2.3). Additionally, the experimentations were extended to find the best parameter values for each algorithm. The results of the experiments are shown in Table 7.3.

For the Community-Based Graph Clustering, we used the NetworkX[44] library in Python, which allows us to create weighted graphs. The clusters are then built from the graph by optimizing the modularity. For the text representation used by the graphs, we used the text embeddings derived by combining GLOVE and LDA. For the weight of the edges, we define the distance matrix via HOTT re-weighted with Cosine similarity. In the end, we had a 4023 $\times$ 4023 sized distance matrix as we had 4023 news articles as BLC from the validation set to be used further as training data in the pipeline's second phase. For the weighted graph, this meant 4023 nodes and, for each node, 4022 edges, which was a pretty complex and cluttered network to comprehend.

For this purpose, we wanted to select top k edges based on distances between the nodes and subsequently use the distance for thresholding the best edges. To achieve the same, we sorted the distance matrix in ascending order such that for every document, we have the distances sorted from most similar to most dissimilar document and then take the average of each level (cf. Figure 7.1). This gives us the average distance between the most similar and dissimilar documents. Finally, we use this distance and their level to get our experiments' best value of k.

[44] https://networkx.org/

Embedding	Algorithm	Similarity	Silhouette Score	Calinski Harabasz	Davies Bouldin
TF-IDF	k-Means	Euclidean	0.0003	17.70	13.41
Doc2Vec	k-Means	Euclidean	0.1231	691.45	2.25
TF-IDF	DBSCAN	Euclidean	0.0110	89.29	1.14
Doc2Vec	DBSCAN	Euclidean	-0.2480	4.49	1.76
TF-IDF	Affinity Propagation	Euclidean	-0.0030	5.47	1.80
Doc2Vec	Affinity Propagation	Euclidean	0.0790	257.49	2.14
TF-IDF	Mean Shift	RBF Kernel	**0.5100**	909.26	0.89
Doc2Vec	Mean Shift	RBF Kernel	0.1600	250.52	2.44
Doc2Vec + PCA	Mean Shift	RBF Kernel	0.3410	2995.94	1.17
TF-IDF	Spectral	RBF Kernel	0.0010	16.79	14.56
TF-IDF + PCA	Spectral	RBF Kernel	0.4410	4066.74	0.86
Doc2Vec	Spectral	RBF Kernel	0.1230	570.49	2.10
Doc2Vec + PCA	Spectral	RBF Kernel	0.3880	**4314.134**	0.86
TF-IDF + PCA	HDBSCAN	Euclidean	-0.1570	383.25	2.68
Doc2Vec	HDBSCAN	Euclidean	-0.1950	21.60	2.83
Doc2Vec	Community Graph (Thresholding)	Cosine	0.1000	608.64	2.18
Doc2Vec	Community Graph (No Thresholding)	Cosine	0.1150	678.45	2.28
GLOVE + LDA	Community Graph (Thresholding)	HOTT + Cosine	**0.5100**	3998.96	**0.73**

Table 7.3: The settings of various clustering techniques are compared here using different combinations of Embedding, Algorithm and Similarity measures. Here the thresholded community-based graph clustering performed the best in combination with GLOVE and LDA-based topic embeddings, and HOTT was re-weighted with Cosine Similarity. For this setting, we had the highest Silhouette Score, indicating more cohesion within the groups and better separation between different groups. The lowest Davies Bouldin score indicates a better separation between different groups.

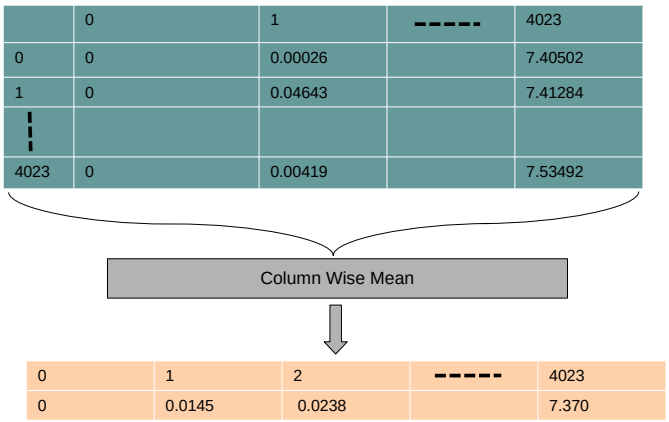


Figure 7.1: The figure shows the conversion of the distance matrix of 4023 × 4023 into a compact 1 × 4023 matrix wherein each column represents the average of the distance at a particular level. Here levels are defined based on the average distance between the most similar documents (level-1) and that of the most dissimilar documents (level-4023)

For the experimentation, we plotted a graph, as seen in the Figure 7.2, with levels 1 to 4023 (most similar to most dissimilar) on X-axis and average distances at each level on the Y-axis. In the first subgraph, we see a trend across each level and certain jumps in the distance between levels 1 and 500. We then zoom up between levels 1 and 500 and see four jumps in the distances between levels 100-150, 200-250, 400-450, and 450-500. The first jump is not that much, so we wanted to experiment with the jumps between 200 and 500 and made a zoomed-up graph between levels 200 and 500. The first jump was between 230 and 240, and the second was between 400 and

440. So we took the value 330 as k since it lay between the two jumps and would give us the idea of how similarity changes across different levels.

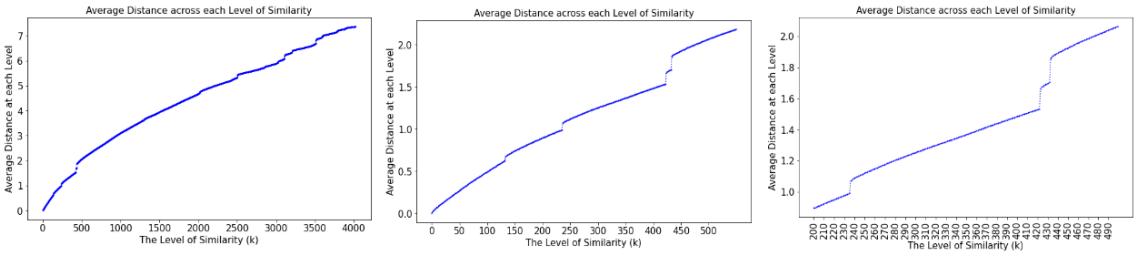


Figure 7.2: The figure depicts zoomed-up views of experimentation between the different levels of similarity and average distances at each level in order to find the top k similar articles for each news article. This is done to reduce the number of edges from each node in our graphical representation of articles wherein the nodes are the articles and are connected to a similar article via the edges weighted by the distance between them.

Now we had a distance for the most similar 330 articles for every article, which were used as the weight of the edges between the nodes (articles). We also experimented with the other k values and evaluated the graph clusters based on modularity, which was *0.84* when $k = 330$. Modularity was chosen as the evaluation measure because the partitions (groups) in the graphs were made using the Louvain Heuristics that try to make the partitions based on the maximum modularity (cf. Section 2.2).

7.3 High-Level Descriptors: Tools, Setup, Results and Evaluation

As discussed in Section 7.2, the BLC from the validation set used in the first phase of the pipeline was divided into smaller groups of articles with similar patterns. These group numbers were used as S-HLD. Also, these groups provided us with different chunks over which the other HLD were derived (c.f. Section 5.4, 5.5.2 and 5.5.1). To summarise the setup, in the second phase of the pipeline, training data was the BLC from the validation set of the first phase, i.e., 4023 news articles. The testing was done using the BLC of the test data from the first phase, i.e., 4076 articles. The flow can be seen in Figure 7.3.

7.3.1 External Tools and Setup

In the pipeline's second phase, most features were handcrafted with the aid of already existing tools and libraries. For example, we used the functionalities of WordNet for the features, making use of the *hypernym-hyponym* structure, and for the custom embeddings, making use of the POS categories. The WordNet functionality offered by the NLTK library[45] allows us to find the word's hypernyms and hyponyms while using the synsets of the word (c.f. Section 2.7).

In order to make sure the synsets of the words are contextual, we use the lesk algorithm, which is part of the NLTK library in Python. A word is labeled as *Specific*

[45] https://www.nltk.org/howto/wordnet.html

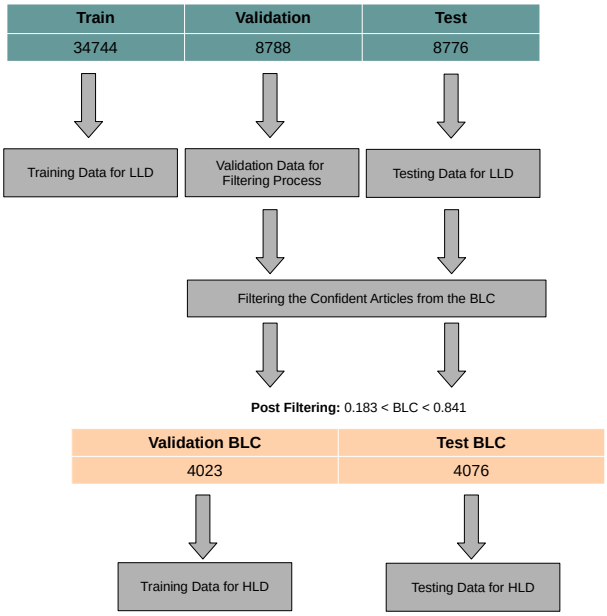


Figure 7.3: The figure summarises the path followed by the Fake News Corpus to reach the final decision. We start with all the articles in the data for the LLD and then filter out the confidently predicted articles and only analyze the Border Line Cases for the HLD. This results in faster computations since HLD requires high computational power, which increases with the number of articles to be considered.

if there are no hyponyms and multiple hypernyms associated with it. The word is labeled as *In-Between* if there are hyponyms and multiple hypernyms associated with it. Finally, the word is categorized as *General* in all the other cases. For the custom embedding categories, we use WordNet's metadata and the mechanism described in Figure 5.6 and 5.7.

For the features attribution, topics, sentiment analysis, and entailment, we used different deep learning models and embeddings described in Table 7.4. Additionally, for attribution purposes, we used non-contextual embedding for comparison purposes using the Doc2Vec model, wherein we trained the Doc2Vec model for our experiments on the labeled article pool (D). This pool is a subset of our fake news corpus, which contains around 50,000 articles on politics, sports, and lifestyle topics. We used the gensim[46] library for Doc2Vec. For our experiments, we used an embedding with a vector length of 100.

RoBERTa embeddings with 1024 dimensional dense vectors were used as features to act as baseline comparisons to our derived HLD. They were derived using sentence transformers, for which the base model was chosen as *nli-roberta-large*. We chose RoBERTa as the baseline because it has been proven as a strong baseline in the field of semantic search.

7.3.2 Results and Evaluation

Once the HLD were derived, the classifier in the second phase of our pipeline (*Classifier-2*) was used to compare the baseline and the proposed features. Like in LLD, we

[46] https://pypi.org/project/gensim/

Feature	Granularity	Hugging face Model
Entailment	Sentence Pairs	roberta-large-mnli
Sentiment	Sentence and Segment	distilbert-base-uncased-finetuned-sst-2-english
Topics (MPQA)	Sentence and Document	microsoft/deberta-v2-xlarge-mnli
Topics (ZSL)	Sentence and Document	facebook/bart-large-mnli
Attribution	Sentence	all-MiniLM-L6-v2

Table 7.4: The table summarises the different hugging face models used with different granularity levels. These models aided in extracting various HLD from the articles, which helped us understand the semantics of fake and real articles.

used various algorithms such as Random Forest, XGBoost, Gradient Boosting, and Decision Tree for the classifier and chose the one that performed best based on the accuracy.

The baseline setup used *RoBERTa* embeddings which resulted in an accuracy of 84.27% with the Random Forest algorithm outperforming the others. We had derived a total of 3061 HLD features, resulting in an accuracy of 83.61% over the test data, with the Gradient Boosting model outperforming the others. Although RoBERTa performed better than HLD, it was at the expense of explainability. The handcrafted 3061 features are easily explainable to the user, as shown in the examples in Section 5.6.2.

Furthermore, to reduce the complexity resulting from the 3061 features, we selected the top 40 features using the Feature Importance weights (c.f. Section 2.5.2) assigned to the features. The Feature Importance ranks the features according to their usability for making boosted decision trees in the model. The more usable the feature is, the better it is ranked. With the top 40 features, the Gradient Boosting resulted in an accuracy of 81.40 %. The reduction of the features decreased the complexity of our system and enhanced the explainability as we have less number of features to analyze. The results are summarized in Table 7.5.

Features	Number of Features	Model	Train Accuracy	Test Accuracy
Roberta	1024	Random Forest	88.21%	84.27%
HLD	3061	Gradient Boosting	86.67%	83.61%
HLD reduced	40	Gradient Boosting	86.10%	81.40%

Table 7.5: Post threshold, we made use of BLC (4023) of the validation set as training data to experiment with the second phase of pipeline making use of HLD. The baseline used here was the RoBERTa embeddings. We achieved an accuracy of 83.61% using our approach compared to the baseline accuracy of 84.27%.

Summarising the performance of our system, let us say we have n articles as confidently predicted articles in the first phase of the pipeline with accuracy a_1 and BLC b are then sent to the second phase of the pipeline. These BLC are then predicted with the accuracy a_2, then the overall performance of our proposed approach a_3 is as follows:

$$a_3 = \frac{a_1 \cdot n + a_2 \cdot b}{n + b} \tag{7.1}$$

Plugging in the values in the Eq. 7.1 as $n = 4700$, $a_1 = 93.08\%$, $b = 4076$ and $a_2 = 83.61\%$. The overall accuracy of our system came out to be **88.68%**.

To compare it with a system without any pipeline system, we used RoBERTa embeddings on the complete dataset to get the features, i.e., trained the classifier using just the RoBERTa embeddings on 34744 articles and tested on the complete 8776 articles of the test set. This resulted in an accuracy of 90.47%. Although this was a better performance in terms of accuracy than our system, it came at the expense of a complete lack of explainability. There was also no way to determine where the system could be improved. This is covered in greater detail in the following section.

We could not provide the exact comparison with the other detection systems made using the Fake News Corpus because no research we came across had the same experimental setup. This means that no one considered the Fake articles to be a combination of both the *fake* and *bias* labels from the Fake News Corpus. Although, by just comparing the experiments conducted making use of the articles under label *fake*, the work by Shrestha [2018] claims to have achieved the accuracy of 87.9% using the *Random Forest* classifier on features such as the Doc2Vec embeddings of the content and title of the news, overall rank of the domain, authors of the news, etc. The work used primarily the metadata about the news articles rather than the textual aspects of the same. In another work by Lorent and Itoo [2019] making use of only the Fake News labels and TF-IDF features claims an accuracy of 89% using Decision Trees as classifiers. The maximum accuracy was achieved in work by Rodríguez and Iglesias [2019] who claim to have achieved the accuracy of 93.7% using the CNN-based model with the input as Word2Vec embeddings.

7.4 Feature, Segment and Clusters Evaluation

7.4.1 Evaluation of Attribution

Although we achieved an accuracy of 84% with the HLD, we wanted to check if we could bring better use of the features which had the highest importance in the classifiers' prediction. According to the Feature Importance score outputted by the Gradient boost model for the HLD, we saw that *attribution* as a feature had a considerable role in the classifier's decision. This prompted us to look into attribution in detail.

We hypothesized that more value could be extracted from this feature if the main labeled document pool (D) contained more articles related to specific topics to which a test query could be attributed. To demonstrate this, we created an additional dataset as a labeled document pool containing the specific keywords *Atom* and *Korea* (cf. Section 4.1.4). These specific keywords were also extracted from the test queries. As a result, we extracted and tested 9570 news articles. In this biased setting, we achieved an accuracy of **92.31%** using only the attribution feature.

This proved that if we had a sufficient number of articles on topics similar to the topic of the test query to which the label of the test query can be attributed, we would have better results. However, of course, one can argue that it also introduces biases in the dataset concerning specific domains. For this reason, we did not use this dataset for further experiments and instead continued working with the unbiased fake news corpus, which also affected the attribution feature performing poorly.

7.4.2 Evaluation of Text Segmentation

The text segmentation, as seen in Section 5.3.1 was done to analyze the text at different levels of granularity. To do so, we evaluated the HLD at sentence and segment levels to check if we could get more information at a particular level of granularity. However, both the sentence and segment level resulted in the same decision-making power of the classifier. Hence, we decided to keep only the features at the sentence level since the time and complexity of the sentence as a feature is much less than that of the segment level. Segments indeed helped us get the partitions at the topic level, but that aspect is then covered by extracting topics as features (cf. Section 5.4.2)

7.4.3 Evaluation of Clusters

We briefly introduced in Section 5.3.2 that the groupings post clustering were used both as an S-HLD for the *Classifier-2* to make the decision and also used as a way to partition the news articles into smaller chunks over which the HLD were derived.

To start with the within-group analysis, we analyzed how the articles were distributed across every cluster. The final clusters using Community-Based Graph clustering can be referred to in Figure 7.4. The clusters 6,10, and 11 were dominated by the real articles and can be seen together in 2-D space as seen in the left part of Figure 7.4. On the other hand, clusters 1,7,8, and 14 were dominated mainly by fake articles and formed a close grouping, as seen in the right part of Figure 7.4.

We further analyzed the word usage of the words in the clusters dominated by fake and real articles. For the clusters 6,10 and 11 (mostly real), we see in Figure 7.5 that cluster 6 and 10 mostly talk about the *policing* matters such as *police*, *official*, *killing* and *attack*. The words mostly indicate the domestic matters of an area and hence are less likely to capture the wider audience (meta goal of Fake news). It might be one of the reasons such news articles are mostly real, as posting fake content on domestic issues would not tend to gain much attention. Cluster 11 on the other hand talks about *medical* and *sports* domain. One reason the articles are real can be that medical news requires domain expertise and a specific set of jargon to describe a situation, and hence is difficult to generate fake content about the same.

The word usage of the clusters dominated by fake texts, namely clusters 1,7, 8, and 14, can be seen in Figure 7.6. The clusters 1,7 and 14 are dominated with the words relating to *politics* such as *President*, *Trump* and *Obama*. As seen in Section 3.1 politics is the hot topic for fake news because of its large impact on society. Also, the political articles are biased towards a particular candidate. On the other hand Cluster 8 talks about *Business* with words such as *money*, *bank*, *market* and *economic*. It is

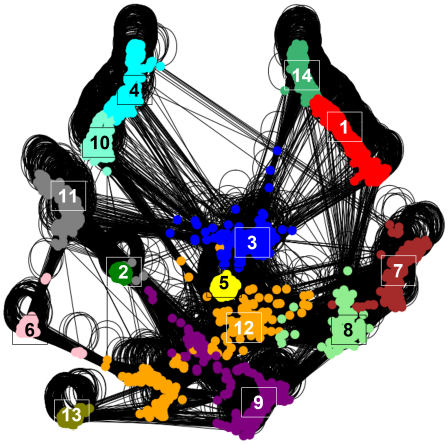


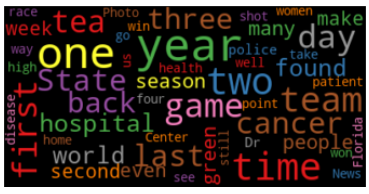
Figure 7.4: The figure shows the resulting groupings from the graph clustering with HOTT re-weighted with the cosine similarities as the weighted distance between the edges. We had 14 groupings, with most groups having a balanced distribution of real and fake news articles with a few exceptions. The exceptions where the real articles dominated were groups 6,10,11, which contained 77%, 64%, and 62% of real articles, respectively. The groups where the fake articles dominated were 1, 7, 8, and 14 with 65%, 68%, 66%, and 69% fake articles, respectively.

usually seen that where there are higher stakes (money in this case), we mostly see the influx of fake news content.

Next, we wanted to see if some groups are easier to predict than others. Furthermore, we wanted to check if predicting different groups separately increased the power of the classifier to distinguish between the fake and real articles at an overall level. To do so, we used *Classifier-2* separately on each of the groups, and the results of the same are given in Table 7.6. We found out that evaluating separately on the groups did not aid in the classifier decision, with the average accuracy from all the groups being approximately 79%.

(a) This sub-figure shows the top 50 words representing all the articles in Cluster 6

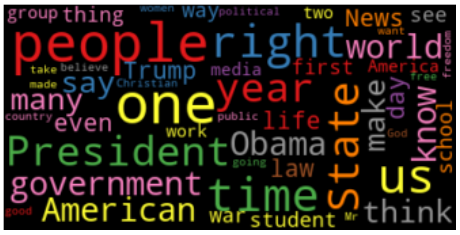
(b) This sub-figure shows the top 50 words representing all the articles in Cluster 10



(c) This sub-figure shows the top 50 words representing all the articles in Cluster 11

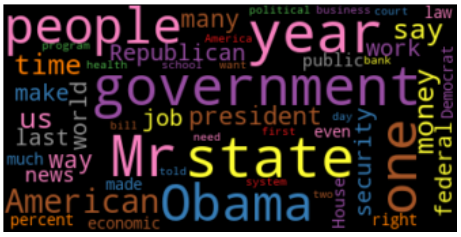
Figure 7.5: The figure depicts the word cloud representing the top 50 words of the groups 6,10, and 11 as seen in Figure 7.4. These groups have the majority of real articles, and we wanted to analyze what kind of words dominate the real article groupings.

(a) This sub-figure shows the top 50 words representing all the articles in Cluster 1



(b) This sub-figure shows the top 50 words representing all the articles in Cluster 7

(c) This sub-figure shows the top 50 words representing all the articles in Cluster 8



(d) This sub-figure shows the top 50 words representing all the articles in Cluster 14

Figure 7.6: The figure depicts the word cloud representing the top 50 words of groups 1,7,8, and 14 as seen in Figure 7.4. These groups have the majority of fake articles, and we wanted to analyze what kind of words dominate the fake article groupings.

Cluster Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Accuracy	74.58%	82.02%	69.34%	73.92%	76.65%	77.76%	82.00%	81.73%	75.79%	70.40%	79.04%	73.43%	85.97%	80.24%

Table 7.6: The performance within the groups was analyzed to see if it enhances the overall performance and also to check if some groups are harder to predict than others. We see that average performance is approximately 79% and so evaluating as a whole is a better setting for our approach.

8. Discussion

In this penultimate section of our thesis, we are performing a qualitative evaluation of our system FACADE. We will put it into the context of the original goals of the thesis. For that, we are analyzing first how our model performs on individual examples that are either (a) correctly predicted by all the algorithms, including the baseline TF-IDF, (b) that are mispredicted by our system and all other models, and (c) that are only correctly predicted by our system. After that, we will check the performance of our system under different thresholds and explain how this would affect an end user in Sectiob 8.2. Last but not least, we will show how the explainability components used in our system will help such an end user in determining whether a news article is really fake or not (cf. Section8.3).

8.1 Analysis of individual news articles

8.1.1 Correctly classified articles

First, we are analyzing articles that all models correctly classify. One such article is *Research in Motion Coming into Double Bottom Support Level*[47] from https://inthemoneystocks.com/. It is a fake news article that TF-IDF classified it 65.2% probably fake. Our system was much more confident with 87.4% probability. The main influencing factors in our system were that only very few sentences were attributed as real (also by doc2vec) and that the noun category, nouns denoting attributes of people and objects, often appeared inside the text. This indicates that the text might be more subjective, especially about people instead of facts. I.e., in this case, the text is about some rising stock, but the only proof is some chart (which might be made up) and the fact that it was written by a *Chief Market Strategist*, which also may not be the case.

In another article, *Before the Summer Buzz Fades Away*[48] from https://www.nytimes.com/, which is a real article and all classifiers predict it as real. TF-IDF was

[47] https://inthemoneystocks.com/blog/2010/08/20/13371-research-in-motion-coming-into-double-bottom-support-level/

[48] https://www.nytimes.com/2007/08/31/arts/31last.html

sure it was real with a probability of 87%, and our system even gave it a 99.55% chance to be correct. Our system already flags as confidently real after only seeing the low-level features, such as the words theater, museum, season, art, and music from TF-IDF, but also after seeing the amount of names appearing in the text, the amount of punctuation, and overall length of the text. The number of noun phrases and the relatively low type-token ratio push it slightly towards being fake, but overall this does not affect the decision much. The latter is probably because it is a mixed article about different parts of culture like a movie, an exhibition, or a play in the theater. TF-IDF probably finds these words necessary since the amount of fake content inside those categories is marginally low because misguiding people on art and music is not as profitable as in other areas like economics and politics.

8.1.2 Wrongly classified articles

Secondly, we are analyzing articles that are detected incorrectly by all classifiers. One such article is *Could Al Franken Lose in Minnesota?*[49] from https://www.nationaljournal.com/, which already has been deleted from the web, potentially because it is fake. However, it is detected as real with 70.2% probability by TF-IDF and 62.8% probability by our system. In our method, it is primarily because the text's POS structure seems real. There are enough sentences attributed as real by Doc2Vec, especially not many fake sentences found. A fake sentence at the end would have been a potentially strong indicator of it being fake, but it is not detected as such. Additionally, the topic *international politics* was detected, which also points more towards a real document. Just by glancing over the news in question, it is hard to identify as a fake news article and could also easily be seen as real news by a human reader. Therefore, improving fake news detectors like our system is even more critical.

Another wrongly predicted article is *VEGOILS-Market factors to watch Nov 4*[50] from https://finance.yahoo.com/ which is a real article. For TF-IDF, this is a borderline case with 51.4% probability, therefore barely detected as fake. The LLD- based classifier is more confident and says it is fake with 67.31% probability. Our systems claim it is with a 64.4% probability of being fake. This prediction can be seen as partially borderline since many sentences are being attributed to being real, but on the other hand, Doc2Vec has way fewer real sentences it tags. Additionally, the structure of the POS tags resembles that of a fake news article. Additionally, the last sentence ends on a fake attributed sentence. If we look at the article, we can understand why the structure might be seen as fake due to its bullet point nature. Additionally, as seen in the beginning, we had an article about the stock market, which was fake. Therefore there are strong attributions toward the fakeness of such stock market articles. It might be more challenging for a classifier to detect whether such articles are valid information; therefore, a pure automatic system on such content might not be that easy. This could also be a potential approach in deciding which content a fact-checker has to screen and which not, but we did not investigate further.

[49] https://web.archive.org/web/20140828205344/https://www.nationaljournal.com/politics/finding-al-franken-20140828/

[50] https://finance.yahoo.com/news/vegoils-market-factors-watch-nov-002117683.html

Additionally, the article above observed an interesting behavior in our system. The prediction probability will change if we manually copy the text and headline from the given link. The classifier making use of LLD will become a piece of borderline real news with 60.52% probability, and our system denotes it barely as fake with 52.28% probability. This might be due to how the news article was cleaned in the dataset we have used. It also shows how much influence proper preprocessing has on the final result of the classification. The reasons for deciding it is fake remain the same, though, and are consistent with the explanations given above.

8.1.3 Articles only correctly identified by FACADE

Last but not least, we are analyzing articles that were only correctly identified by our system. The article *Woman shouts slogan against PM Modi at Bhatinda rally; cop says minister abused him*[51] from https://indianexpress.com/ is actually true. TF-IDF classified it as fake with 58.8% confidence, and the LLD level classifier saw it 76.05% as fake. LLD mostly saw the number 2017 as a problem, which might indicate that fake news often talks about years. Additionally, the headline length was deemed too long for real news, and the polarity of the text was also too negative. Our whole system correctly classified the news article with a 91.2% probability. This is mostly because many sentences were attributed to being real, very few only as fake. The POS Structure also is classified as real. It does not end with a fake sentence as well. Therefore it is shown how important the attribution feature is.

Kasich struggles to win public approval for budget cuts[52] from https://beforeitsnews.com/ is another article that wrongly predicted by all other classifiers except our system. It is a fake news article, but TF-IDF claims it is real with merely 62.8% probability, whereas the LLD level classifier says it is real with 70.1% probability. This is primarily due to the length of the article and the existence of the word percent inside it. On the other hand, LLD shifts it towards a fake side due to the low number of noun phrases and the high subjectivity of the text. The high subjectivity actually might be a good indicator of this being fake since people and their opinions mainly write the source website. Our method derives it as fake with 77.4% probability. This is due to less number of sentences being attributed as real. Additionally, the POS Structure points toward fake writing. The topic *politics economy* was adequately detected. Along with the number of fake sentences being attributed, it was deemed partially real, which is why the probability of it being fake is not that high.

Overall, we can see that our system works as intended but is, of course, as such not perfect yet. Some attributes push the classification in one direction, therefore overpowering other decision criteria. This can be seen, for example, in the POS Structure, where a writing style that resembles more closely that of fake news automatically pushes the decision firmly towards fake content, even though the attribution gives it high credibility. Some topics might also need special care since they do not behave well with the attribution feature and might be overly eager to claim an article is fake or real. Still, our attribution feature works well enough in

[51] https://indianexpress.com/article/india/india-news-india/woman-shouts-slogan-against-pm-modi-at-bhatinda-rally-cop-says-minister-abused-him-4395451/

[52] https://beforeitsnews.com/tea-party/2011/03/kasich-struggles-to-win-public-approval-for-budget-cuts-511362.html

distinguishing between fake and real news and is often the number one criteria for deciding that. We also saw that some basic statistical features could already properly detect specific news patterns. Therefore we have shown that our chosen methods are used for detecting fake news (cf. **RQ-1**) and also how some of the factors we extracted influence the decision of the system (cf. **RQ-2a**). Additionally, even though we did not find that any of the extracted clusters had any substantial influence on the prediction quality of our system, there seem to be some topics inside the dataset for which the decision is leaning more towards fake (e.g., economics) or real (e.g., arts). We especially identified the stock market as a potentially tricky topic to detect real news for, therefore partially answering **RQ-2b**.

8.2 Application for a content manager

Additionally, we want to discuss the usability of our system FACADE in the context of tagging news articles for a content manager or fact-checker (as described in the Introduction, cf. Section 1). For that, we analyzed the behavior of our system for different thresholds in the filtering step of the cascade. This behavior we quantified using Accuracy, False Negative Rate (FNR), and False Positive Rate (FPR) in Table 8.1. We additionally show how many documents are left over for review to the content manager, which the method deemed borderline cases. As a reminder, the *fake threshold* (t_1) at this moment denotes the probability that the classifier's output has to be below for a document to be deemed confidently fake. On the other hand, the *real threshold* (t_2) provides the value the classifier's prediction has to be over for documents to be classified confidently as real. Finally, all others will be left over for review.

We specifically chose to select FNR and FPR since the former shows how much real news is predicted as fake, whereas the latter provides us with the number of fake news detected as real. Depending on the setting, either might be a more significant challenge. I.e., if the FNR is high, it would mean that much real news is being filtered out as fake, which can be seen as a kind of censorship and would therefore be frowned upon. On the other hand, a high FPR would mean that more fake news is deemed real, thus keeping those on the platform and possibly spreading further. The problem is that it also makes no sense to give all the work again to the fact-checker and wait for their review to come in. Therefore, optimally, we want to optimize a scenario that best fits the business use case. I.e., is a more robust filter for fake news more advisable, whereas wrongly flagged real news can be challenged by its owner, or should the platform be relatively more open and therefore have a less strong filter.

If we look at the results in Table 8.1, the complete automatic tagging of all news is seen in instance 1, which would mean no additional work for the content manager, but it would misclassify almost 19% of the news that is fake as real and vice versa. Therefore this kind of system would both be censoring as well as still spreading too much fake news as real. On the other end of the spectrum is entry number 9, where the maximum number of news is still left for the content manager to review (2813 out of 4076, so almost 70%), but the FNR and FPR are both low. Realistically speaking, though, removing only 30% of the news overall is much too low to be sensible for a real-life system, especially if we would assume the scanning of social media like

	Threshold					
No.	Fake	Real	Accuracy	FNR	FPR	News for review
1	0.5	0.5	81.40%	18.79%	18.60%	**0** (0.00%)
2	0.5	0.62	83.04%	20.45%	13.58%	284 (6.97%)
3	0.5	0.9	84.22%	41.96%	**1.76%**	1471 (36.09%)
4	0.389	0.5	83.25%	13.72%	19.73%	236 (5.79%)
5	0.389	0.62	85.26%	15.02%	14.45%	520 (12.76%)
6	0.389	0.9	87.67%	33.20%	1.89%	1707 (41.88%)
7	0.1	0.5	85.33%	**1.19%**	35.70%	1342 (32.92%)
8	0.1	0.62	88.48%	1.33%	27.64%	1626 (39.89%)
9	0.1	0.9	**96.04%**	3.60%	4.11%	2813 (69.01%)

Table 8.1: The table gives an overview of the Accuracy, False Negative Rate (FNR), False Positive Rate (FPR), and the number of cases not being classified confidently by the system (Documents for review). In addition, the table shows the relation between the amount of work that still needs to be manually done by a content manager or fact-checker versus the goodness of the predictor.

Twitter, where 70% leftover tweets would still mean that 350 million tweets have to be reviewed daily[53] (cf. Section 1).

A more likely setting might be setting number 3, where the FPR is merely 2%. This would indicate that only a few fake news articles are deemed real. The false negative rate is exceptionally high, though, which might lead to many complaints, which would also have to be reviewed. Additionally, 36% of the news still needs to be checked by a content manager anyways. Finally, in setting 7, we have the exact reverse setting, where the censorship is very low, but much fake news is missed and tagged as real. Here also, over 30% of news still have to be reviewed.

That is also why it is so essential that an automatic filtering system is understandable and explainable. I.e., setting number 1 could be feasible if we give all tags sound reasoning so that a decision can be quickly understood and potentially corrected or revoked. Similarly, we could argue for setting number 9, where much news is still left for review. Then, if these borderline cases are still being tagged, but quick reasons are given for deciding whether it is fake, a content manager might be able to sift through these news faster than before. In either case, it would reduce the workload for such a position. Therefore we will discuss this thesis's explainability in the next section.

8.3 Explainability

As explained in the concept in Section 5.6.2, we use the ExplainerDashboard to give an overview of how our system behaves across the whole dataset as well as individual instances (among others). Additionally, we have given some explaining information in the UI, i.e., the sentences attributed to being fake or real, the four most important features used for classification, and a three-level scale of importance. With the latter,

[53] https://www.internetlivestats.com/

we mostly want to give an end user, who is not an expert and does not have to do this professionally, the possibility to check the news for their reliability. Moreover, it can give them hints about parts of the news to look out for and which might be trustworthy. On the other hand, for this section, we focus on a professional who has to check the news (and related entries) for a living, and therefore we assume that they might need a deeper insight into the system's workings.

There are many different things that such a professional (e.g., content manager, fact checker) could investigate in the dashboard. One of the first interesting areas is the overall behavior of the model in terms of the features used for classification. This is being shown in Fig. 8.1. We restricted the figure to only showing the ten most important features. These important features were calculated using the average of all SHAP values across the whole test set. Based on this, it is possible to observe that the two most important features are attributing sentences to real and fake articles. Additionally, it is important whether the last sentence is real (No. 3) or fake (No. 6). As has also been observed in the examples examined above (cf. Sec. 8.1), the structure of the text based on POS tags (denoted here as *markov_pred*), i.e., whether it is more similar to fake or real articles, is also important. Additionally, two topics play an important role: International Political and Domestic Political (No. 7 and No. 10, respectively). Feature No. 5 is a combination HLD that combines the topic of entertainment with the question of whether the last sentence is attributed to a fake article. The two leftover features, No. 8 and 9, are similar to the first two features, except that the attribution is done via doc2vec instead of DeBERTa.

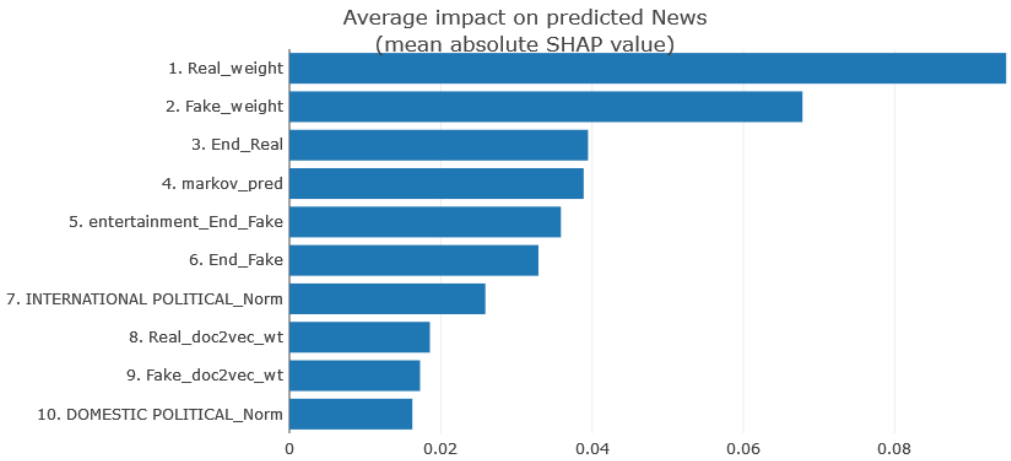


Figure 8.1: This figure shows the ten most important features of the dataset in order to distinguish between fake and real news.

This information provides only a general idea about how the model works and what features it mainly uses. However, it does not give insight into how exactly an individual document was classified. For that, we can make use of individual feature importances per article. For example, consider news titled *Errant Vehicle Kills 3 Women Outside Albany*[54], which is correctly predicted as real by our system. We

[54] https://www.nytimes.com/2011/08/11/nyregion/errant-suv-kills-3-in-parking-lot-near-albany.html

can see in figure 8.2 that here the behavior of our system is similar to the general behavior, i.e., the attribution to fake and real sentences are the most crucial decision criteria along with the POS structure of the text being real. It is also not a text about entertainment where the last sentence ends on a fake attribution. The news itself is about an accident, so not being about entertainment is right. It mistakenly takes up the topics of sports and environment, though (which can not be seen on this figure), probably due to the victims being part of a sports group, and the vehicle that caused the accident is called a *Highlander*. Nevertheless, it still gives a good impression of how the decision may have been made. The article could be identified as actually real, along with the display of the attributed sentences, as seen in the UI section of this thesis (cf. Chapter 6).

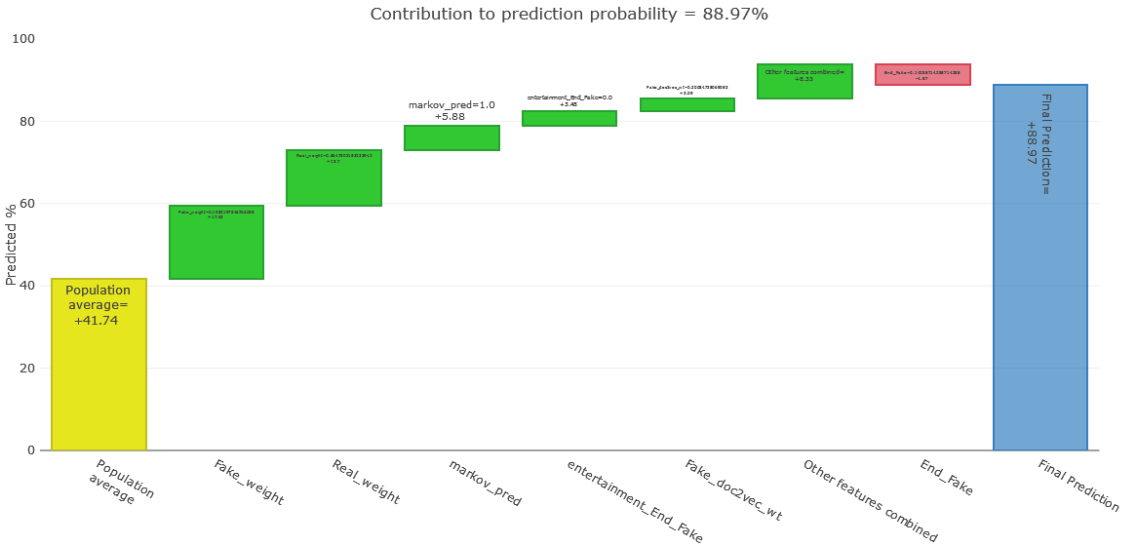


Figure 8.2: This figure shows the six most important features that led to the decision for the news titled *Errant Vehicle Kills 3 Women Outside Albany*. The news is correctly predicted as real.

Another entry is titled *Isreali forces shoot dead Palestinian man in northern West Bank*[55] and is a fake news article. It is a highly biased article, probably trying to rile up people against Isreal as a state. Our system decided that this news must be real with 78.3% probability, as seen in figure 8.3. Here the primary decision criteria are quite different, though, with the main one being that the article is written in a style of a real article. This is true, and the article reads like an actual news article. Additionally, the article is detected as being about the environment and health, whereas the sentence preceding the health topic must be real. Of course, the article is neither about the environment nor about health. However, it is probably detected as the latter due to the text talking about critically injured, wounds, and so on. Additionally, very few sentences are only attributed to being fake, and most text is written neutrally. Therefore, the text's bias is not seen in some form of expressive writing. Instead, the text reads like regular reporting instead of a made-up story.

[55] https://web.archive.org/web/20160829121057/http://www.presstv.com/Detail/2016/08/26/481771/Israeli-forces-shooting-Palestinian-man-death-Silwad-West-Bank

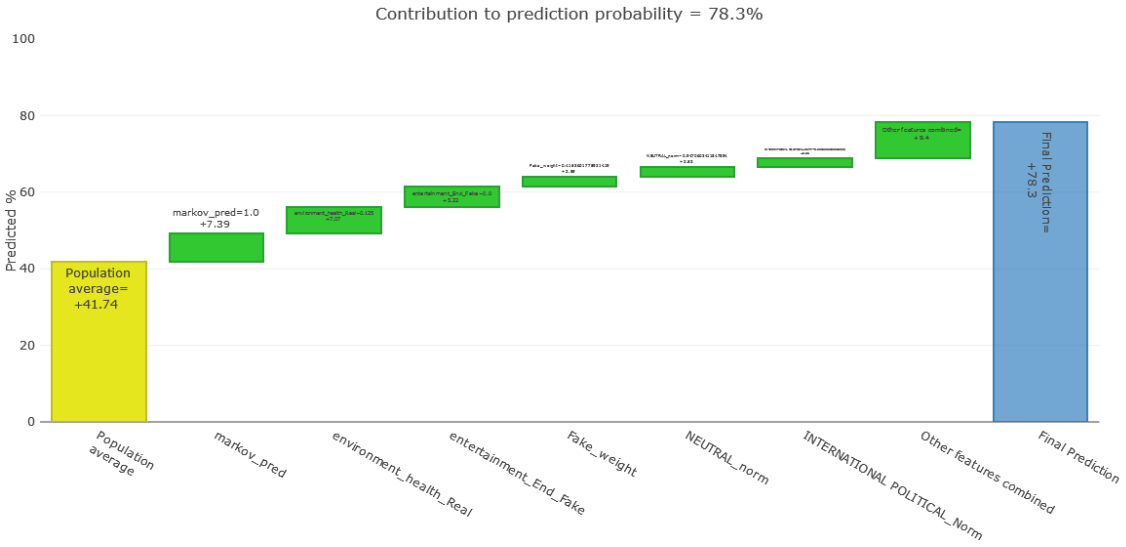


Figure 8.3: This figure shows the six most important features that led to the decision for the news titled *Isreali forces shoot dead Palestinian man in northern West Bank*. The news is a piece of extremely biased news and therefore considered fake. It is mistakenly predicted as real by our system.

Last but not least, we looked at news called *Full text: PM Narendra Modi's speech at CII-Keidanren Business Luncheon in Tokyo*[56], which is a real article but is being detected as fake with 62.6% probability. The contributions to this wrong prediction can be seen in figure 8.4. Firstly, it is correctly detected as a political topic, which brings the news article towards real classification. Also - partially hidden in the *Other features combined* - the topics environment and sports were picked up again. The former is a correct detection for this news since in the speech, the PM of India talks about cleaner energy and the environment in economics. Additionally, the total number of real sentences pushes it towards being real, along with the fact that it ends on a real sentence, not a fake one. Still, as can be seen in the figure, these features are entirely outweighed by (a) the overall bias of the classifier towards predicting news as fake first (population average is below 50%) and (b) the other three essential figures depicted in the graph. Moreover, these are the weighted fake and real sentence occurrences (attributions), and the writing style (POS tag structure) is more closely related to fake news than to real ones. The latter might be the case due to this news article citing a speech that might be more opinionated and straightforward to appeal to the people. Therefore it might even be true that not all statements made in the speech are entirely correct.

Of course, this explainability could be made more understandable by, for example, providing explanations for every feature and potentially visualizing some of these features as we did for the attribution feature. Still, the current system gives a good idea of how our system can be transparent to a user and how it already provides understandable features. If these features can be even more reliable, the agreement to

[56] https://indianexpress.com/article/india/india-news-india/full-text-pm-narendra-modis-speech-at-cii-keidanren-business-luncheon-in-tokyo-4369369/

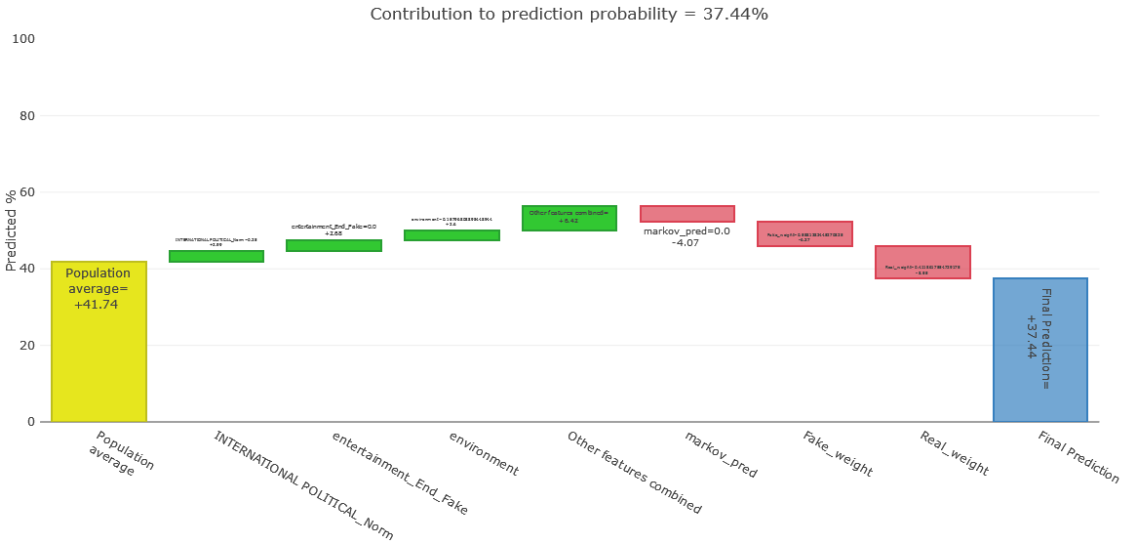


Figure 8.4: This figure shows the six most important features that led to the decision for the news titled *Full text: PM Narendra Modi's speech at CII-Keidanren Business Luncheon in Tokyo.* The news is real and is incorrectly predicted as fake.

the explanations might also be high. We have shown how for example, the attribution feature can be made more reliable by having a closer corpus to the use case, as seen in Section 7.4.1. Overall, though, we showed in this section that our chosen methodology of explainable features combined with posthoc explanations could be used to make a fake news detection system transparent, thus answering **RQ-3**.

9. Conclusion

9.1 Summary

This thesis presented a system called FACADE for automatically tagging news as either fake or real. Our goal was to provide, on the one hand, an explainable classification and, on the other hand, a distinction of news according to their difficulty of prediction. We achieved the latter by designing our system as a cascade, meaning we are splitting the classification task into different levels. Therefore, we had a first level, where we only employed basic features like the length of the text, Type-Token-Ratio, grammatical mistakes, and so on, to filter out already news that is correspondingly easy to classify. Then, using some thresholding mechanism, we take over the less confident documents into the second cascade, where the features are more high-level, e.g., source-based sentence attributions, structural identification, custom noun, verb, and adjective embeddings. We have shown via that way that not all fake news requires complicated classification algorithms or features and can already be easily filtered out. That way, processing power can be saved, and understandability can be increased. Therefore we have shown that for answering **RQ-1**, different methods are needed to detect fake news reliably and that we can combine them in one large pipeline.

Therefore, we can answer for **RQ-2a** that the methodology for detecting fake news depends on its complexity. We used different levels of features and have shown that these features work well for their respective task. The most promising feature overall was the attribution feature, which tagged sentences as real or fake depending on the similar sentences found in the corpus and the classification of the article the sentence was from. On the other hand, the most important features for the low-level part were TF-IDF-based word features but also features like the ratio of verbs to adjectives, the ratio of nouns to adjectives, the type-token ratio of the headlines, and polarity as well as the subjectivity of the text.

The differences between the cascades already also partially could answer **RQ-2b**, since this showed that there must be certain news that is easier to classify than others. However, we still investigated whether, among the more challenging to classify news,

there might be other groups that potentially behaved differently. For example, we could find groups biased toward real or fake news with different clustering techniques. However, none of the groups stood out in their classification accuracy compared to treating all news similarly. Therefore we could not provide any further insights yet but showed in the discussion in section 8.1 that there might be still some groups inside the dataset, like politics, economics, arts, and so on, that have a different bias inside the system and therefore might have to be handled slightly differently.

For the explainable part of our thesis, we used different components. On the other hand, we are trying to assess feature importance, i.e., those relevant features for classifying either the whole corpus or individual articles, using SHAP values to give a user the features that led to the decision. For that to be explainable, we tried to restrict ourselves to features that were ante-hoc explainable. Other features that were not easily understandable, we tried to make explainable by providing visualization, e.g., the attribution of sentences by marking sentences as green, if they are real attributed, or as red if they are fake attributed. We also used the ExplainerDashboard, giving more advanced users additional insights. However, our primary approach still relied on explainable features and ante-hoc explainability, thus answering **RQ-3**.

We also made use of the explainability components to investigate our system further. We found out how our chosen features interacted - especially the HLDs - with each other. For example, news detected talking about politics and having many fake sentences would most likely be fake. We also could show that there seem to be some correlations between features, like the subjectivity from LLD and the topics from HDL, which we might have missed including in the classification. However, We could still potentially observe thanks to the explanations. Therefore we can add up to **RQ-2a** that individual features are relevant for detecting fake news and their interaction with each other.

We missed some interactions between features, which was also a limitation of our work. These limitations we will discuss in the following section.

9.2 Limitations

Our work consists of several limitations at the moment. For a better overview, we will divide those into technical limitations, i.e., challenges we faced due to a lack of resources or hardware, and methodological limitations, i.e., problems that directly stem from the applied methods and features. We will first have a look at the technical limitations.

Technical Limitations

Our prominent feature is the attribution feature. As shown in section 7.4.1, this feature could have been performing much better if the corpus had had more examples about a particular subject or topic. I.e., we did not have many articles about India in our dataset, or a lot of the news about Russia was fake news. Due to that, the attribution either does not work for those articles (e.g., in the case of India) and returns more or less random attributions or primarily attributes to only one type of class and therefore has an inherent bias (e.g., in the case of Russia). This mainly

was a limitation of our hardware resources, though, since we could not process the entire Fake News Corpus and therefore could not examine the attribution value for a more extensive dataset. We assume we could have achieved much higher accuracy if we had been able to do that.

Another technical limitation is the speed of the system. A single news article takes much time to process through the whole pipeline. This is dependent on the length of the article mainly since, for example, attribution takes time. Therefore, the system cannot yet be put into a real-life scenario, where it would potentially have to scan thousands or more articles per minute. The scaling up of that system is not merely a hardware limitation, though the general processing inside the pipeline would also have to be made more efficient. Therefore our work is more of a proof of concept.

Methodological Limitations

The attribution feature also has some methodological limitations. For example, the main attribution is done via sentence embeddings. The most similar sentence embedding is being searched for, and if it is above our threshold, it will become a candidate. The best candidate is then selected. Therefore, the context of that sentence is mostly not relevant. We are only considering the semantics of the sentence standalone. We have often observed that the attribution is taken out of context, even though the semantic meaning is the same or similar. E.g., a statement made like "Vaccines are not effective" could be considered fake if taken standalone. However, if it is being said in the context of a particular disease, where vaccines do not exist, it might be a correct statement again. We did not investigate this issue further, in any case.

There are also limitations on how we made use of the different features in the system. Due to how our cascade is designed, we do not use the LLDs later on in the pipeline, even though they could, in correlation with the HLDs, help in distinguishing fake news better. For example, one of the LLDs uses POS Tags to filter out how many times a person is being talked about in the text. Additionally, we have an HLD that extracts topics like sports, environment, etc. Potentially, some topics inherently talk about more people; therefore, a hybrid feature could have improved the detection accuracy. I.e., a sports article will talk about more people, so it is not automatically a fake news article. On the other hand, an article about economics should potentially talk more about numbers and companies instead. Since we strictly separated LLDs and HLDs, such a combination could not have been detected.

9.3 Future Work

Many things can be done to or with our system FACADE in the future. First, the UI should be expanded to include more explainability components. Some ideas have already been presented in section 5.6.2 about other types of explanations like the text plot for SHAP on entailments and sentiment. We decided to leave these out because they did not yet contribute majorly to the decision but could still provide an additional overview, especially if the attribution feature would also be extended to contain the context of the sentence. With such visualizations, it might be possible to show the context of a sentence by showing which other parts of the text contributed

to it. The same could be done then on the source sentence, i.e., instead of showing the matched sentence, we additionally will display the context and how strongly it influenced the decision.

Based on the limitations mentioned above, it is clear that we would like to investigate more features and their combinations to make fake news detection even more reliable. Especially the attribution feature is so promising that properly testing it on its effectiveness and reliability would be interesting. For that, finding some heuristics to speed up the computations and find potential candidate matches for our query sentence is necessary. Currently, the approach works like a brute-force approach and is too slow with quadratic complexity. A possible start would be to filter out unlikely matching news articles, e.g., on metadata (same topic?), document embedding similarities, and so on.

We would also like to employ our system in a more realistic environment, i.e., have a stream of news coming in that needs to be flagged. For that purpose, it would be necessary to increase the speed and provide more and better explanations and support a batch mode and a kind of overview in the end that gives a quick and detailed report for all given news articles. This could be compared to systems like plagiarism checkers that allow the submissions of hundreds of documents simultaneously, with each being given a score of its plagiarism level. Similarly, we would show a likeliness for the news article to be fake and therefore allow the content manager to filter those out that they want to have a look at.

Aside from those direct future use cases, it might be possible to employ our features and our system for different data, e.g., to distinguish whether a document was written by the same or a different author. Our LLDs can be seen as kind of first indicators. For example, the POS structure detector could be trained on texts of the same author in order to understand whether the structure of the text is similar. The attribution feature can also be used to attribute similar content and writing style across other documents. Similarly, our features can also be used for something like plagiarism detection. Overall, our system already showed promise and is definitely worth further investigation in those different areas.

Bibliography

Bill Adair, Chengkai Li, Jun Yang, and Cong Yu. Progress toward "the holy grail": The continued quest to automate fact-checking. In *Computation+ Journalism Symposium,(September)*, 2017. (cited on Page 22)

Julie L Adnsager, Erica Weintraub Austin, and Bruce E Pinkleton. Questioning the value of realism: Young adults' processing of messages in alcohol-related public service announcements and advertising. *Journal of communication*, 51(1):121–142, 2001. (cited on Page 20)

Sadia Afroz, Michael Brennan, and Rachel Greenstadt. Detecting hoaxes, frauds, and deception in writing style online. In *2012 IEEE Symposium on Security and Privacy*, pages 461–475. IEEE, 2012. (cited on Page 23)

Swati Aggarwal, Tushar Sinha, Yash Kukreti, and Siddarth Shikhar. Media bias detection and bias short term impact assessment. *Array*, 6:100025, 2020. (cited on Page 19)

Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. Knowledge distillation from internal representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7350–7357, 2020. (cited on Page 17)

Ahmet Aker, Leon Derczynski, and Kalina Bontcheva. Simple open stance classification for rumour analysis. *arXiv preprint arXiv:1708.05286*, 2017. (cited on Page 21)

Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, 31(2):211–36, 2017. (cited on Page 1 and 19)

André Altmann, Laura Toloşi, Oliver Sander, and Thomas Lengauer. Permutation importance: a corrected feature importance measure. *Bioinformatics*, 26(10): 1340–1347, 2010. (cited on Page 16)

Maxim Alyukov. Making sense of the news in an authoritarian regime: Russian television viewers' reception of the russia–ukraine conflict. *Europe-Asia Studies*, 74(3):337–359, 2022. doi: 10.1080/09668136.2021.2016633. URL https://doi.org/10.1080/09668136.2021.2016633. (cited on Page 1)

Faheem Aslam, Tahir Mumtaz Awan, Jabir Hussain Syed, Aisha Kashif, and Mahwish Parveen. Sentiments and emotions evoked by news headlines of coronavirus disease (covid-19) outbreak. *Humanities and Social Sciences Communications*, 7(1):1–9, 2020. (cited on Page 55)

Meital Balmas. When fake news becomes real: Combined exposure to multiple news sources and political attitudes of inefficacy, alienation, and cynicism. *Communication research*, 41(3):430–454, 2014. (cited on Page 19)

Satanjeev Banerjee and Ted Pedersen. An adapted lesk algorithm for word sense disambiguation using wordnet. In *International conference on intelligent text processing and computational linguistics*, pages 136–145. Springer, 2002. (cited on Page 18)

Jody C Baumgartner and Jonathan S Morris. One "nation," under stephen? the effects of the colbert report on american youth. *Journal of Broadcasting & Electronic Media*, 52(4):622–643, 2008. (cited on Page 19)

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003. (cited on Page 14)

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008. doi: 10.1088/1742-5468/2008/10/p10008. URL https://doi.org/10.1088/1742-5468/2008/10/p10008. (cited on Page 12)

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. (cited on Page 15)

Amy G Bryant and Erika E Levi. Abortion misinformation from crisis pregnancy centers in north carolina. *Contraception*, 86(6):752–756, 2012. (cited on Page 20)

Tadeusz Caliński and Harabasz JA. A dendrite method for cluster analysis. *Communications in Statistics - Theory and Methods*, 3:1–27, 01 1974. doi: 10.1080/03610927408827101. (cited on Page 13)

Clay Calvert and Austin Vining. Filtering fake news through a lens of supreme court observations and adages. *First Amend. L. Rev.*, 16:153, 2017. (cited on Page 19)

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016. (cited on Page 15)

Yimin Chen, Niall J Conroy, and Victoria L Rubin. Misleading online content: recognizing clickbait as" false news". In *Proceedings of the 2015 ACM on workshop on multimodal deception detection*, pages 15–19, 2015. (cited on Page 19)

Effie K Chipeta, Wanangwa Chimwaza, and Linda Kalilani-Phiri. Contraceptive knowledge, beliefs and attitudes in rural malawi: misinformation, misbeliefs and misperceptions. *Malawi Medical Journal*, 22(2), 2010. (cited on Page 20)

Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. Computational fact checking from knowledge networks. *PloS one*, 10(6):e0128193, 2015. (cited on Page 23)

Matteo Cinelli, Gianmarco De Francisci Morales, Alessandro Galeazzi, Walter Quattrociocchi, and Michele Starnini. The echo chamber effect on social media. *Proceedings of the National Academy of Sciences*, 118(9):e2023301118, 2021. (cited on Page 20)

Kevin Coe, David Tewksbury, Bradley J Bond, Kristin L Drogos, Robert W Porter, Ashley Yahn, and Yuanyuan Zhang. Hostile news: Partisan use and perceptions of cable news programming. *Journal of communication*, 58(2):201–219, 2008. (cited on Page 19)

D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5): 603–619, 2002. doi: 10.1109/34.1000236. (cited on Page 10)

Nadia K Conroy, Victoria L Rubin, and Yimin Chen. Automatic deception detection: Methods for finding fake news. *Proceedings of the association for information science and technology*, 52(1):1–4, 2015. (cited on Page 21)

Corinna Cortes and Vladimir Naumovich Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 2004. (cited on Page 15)

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. Recognizing textual entailment: Rational, evaluation and approaches–erratum. *Natural Language Engineering*, 16(1):105–105, 2010. (cited on Page 24)

David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979. doi: 10.1109/TPAMI.1979.4766909. (cited on Page 14)

Amber Day and Ethan Thompson. Live from new york, it's the fake news! saturday night live and the (non) politics of parody. *Popular Communication*, 10(1-2): 170–182, 2012. (cited on Page 19)

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. URL https://arxiv.org/abs/1810.04805. (cited on Page 7)

Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000. (cited on Page 15)

Mengnan Du, Ninghao Liu, and Xia Hu. Techniques for interpretable machine learning. *Communications of the ACM*, 63(1):68–77, 2019. (cited on Page 25)

Delbert Dueck. *Affinity propagation: clustering data by passing messages*. University of Toronto Toronto, ON, Canada, 2009. (cited on Page 11)

Andrew Duffy, Edson Tandoc, and Rich Ling. Too good to be true, too good not to share: the social utility of fake news. *Information, Communication & Society*, 23 (13):1965–1979, 2020. (cited on Page 20)

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996. (cited on Page 9)

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S Weld. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74, 2008. (cited on Page 23)

Song Feng, Ritwik Banerjee, and Yejin Choi. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 171–175, 2012. (cited on Page 23)

Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. The Rise of Social Bots. *Communications of the ACM*, 59(7):96–104, 2016. (cited on Page 2)

Eileen Fitzpatrick, Joan Bachenko, and Tommaso Fornaciari. Automatic detection of verbal deception. *Synthesis Lectures on Human Language Technologies*, 8(3): 1–119, 2015. (cited on Page 21)

Evelyn Fix and J. L. Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review / Revue Internationale de Statistique*, 57(3):238–247, 1989. ISSN 03067734, 17515823. URL http://www.jstor.org/stable/1403797. (cited on Page 15)

Paul A Gagniuc. *Markov chains: from theory to implementation and experimentation*. John Wiley & Sons, 2017. (cited on Page 54)

Siva Charan Reddy Gangireddy, Cheng Long, and Tanmoy Chakraborty. Unsupervised fake news detection: A graph-based approach. In *Proceedings of the 31st ACM conference on hypertext and social media*, pages 75–83, 2020. (cited on Page 24)

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus, 2019. (cited on Page 7)

Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *journal of Computational and Graphical Statistics*, 24(1):44–65, 2015. (cited on Page 62)

Louisa Ha, Loarre Andreu Perez, and Rik Ray. Mapping recent development in scholarship on fake news and misinformation, 2008 to 2017: Disciplinary contribution, topics, and impact. *American behavioral scientist*, 65(2):290–315, 2021. (cited on Page 22 and 25)

John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28 (1):100–108, 1979. (cited on Page 8)

Naeemul Hassan, Chengkai Li, and Mark Tremayne. Detecting check-worthy factual claims in presidential debates. In *Proceedings of the 24th acm international on conference on information and knowledge management*, pages 1835–1838, 2015. (cited on Page 22 and 23)

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention, 2020. URL https://arxiv.org/abs/2006.03654. (cited on Page 7)

Marti A Hearst. Text tiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics*, 23(1):33–64, 1997. (cited on Page 14)

Seyedmehdi Hosseinimotlagh and Evangelos E Papalexakis. Unsupervised content-based identification of fake news articles with tensor decomposition ensembles. In *Proceedings of the Workshop on Misinformation and Misbehavior Mining on the Web (MIS2)*, 2018. (cited on Page 24)

Nancy Ide and Jean Véronis. Introduction to the special issue on word sense disambiguation: The state of the art. *Comput. Linguist.*, 24:2–40, 03 1998. (cited on Page 17)

Frederick Jelinek, John D Lafferty, and Robert L Mercer. Basic methods of probabilistic context free grammars. In *Speech recognition and understanding*, pages 345–360. Springer, 1992. (cited on Page 23)

Zhiwei Jin, Juan Cao, Yongdong Zhang, and Jiebo Luo. News verification by exploiting conflicting social viewpoints in microblogs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016. (cited on Page 24)

Marc Jones. Propaganda, Fake News, and Fake Trends: The Weaponization of Twitter Bots in the Gulf Crisis. *International Journal of Communication*, 13(0), 2019. (cited on Page 2)

Jozef Kapusta, Petr Hájek, Michal Munk, and L'ubomír Benko. Comparison of fake and real news based on morphological analysis. *Procedia Computer Science*, 171: 2285–2293, 2020. (cited on Page 57)

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020. (cited on Page 7)

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch, 1975. (cited on Page 23)

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. 2017. *ArXiv abs/1609.02907*, 2017. (cited on Page 21)

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. From word embeddings to document distances. In *ICML*, 2015. (cited on Page 7)

Jey Han Lau and Timothy Baldwin. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368*, 2016. (cited on Page 6)

Michael E. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *SIGDOC '86*, 1986. (cited on Page 17)

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019. URL https://arxiv.org/abs/1910.13461. (cited on Page 17)

Jamy Li and Mark Chignell. Birds of a feather: How personality influences blog writing and reading. *International Journal of Human-Computer Studies*, 68(9): 589–602, 2010. (cited on Page 54)

Jiwei Li, Myle Ott, Claire Cardie, and Eduard Hovy. Towards a general rule for identifying deceptive opinion spam. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1566–1576, 2014. (cited on Page 23)

Xia Liu. A big data approach to examining social bots on Twitter. *Journal of Services Marketing*, 33(4):369–379, 2019. (cited on Page 2)

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. URL https://arxiv.org/abs/1907.11692. (cited on Page 7 and 16)

Simon Lorent and Ashwin Itoo. Fake news detection using machine learning. *A thesis presented for the degree of Master in Data Science, University of Liège*, 2019. (cited on Page 77)

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017. (cited on Page 62)

Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. Detect rumors using time series of social context information on microblogging websites. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1751–1754, 2015. (cited on Page 24)

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. Detecting rumors from microblogs with recurrent neural networks. 2016. (cited on Page 24)

Bill MacCartney. *Natural language inference*. Stanford University, 2009. (cited on Page 16)

Amr Magdy and Nayer Wanas. Web-based statistical fact checking of textual documents. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pages 103–110, 2010. (cited on Page 23)

David Malvern, Brian Richards, Ngoni Chipere, and Pilar Durán. *Lexical diversity and language development*. Springer, 2004. (cited on Page 23)

William C Mann and Sandra A Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text-interdisciplinary Journal for the Study of Discourse*, 8(3):243–281, 1988. (cited on Page 23)

Philip J McCarthy and Cecelia B Snowden. The bootstrap and finite population sampling. 1985. (cited on Page 15)

Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.*, 2(11):205, 2017. (cited on Page 9)

Phayung Meesad, Pudsadee Boonrawd, and Vatinee Nuipian. A chi-square-test for word importance differentiation in text classification. In *Proceedings of international conference on information and electronics engineering*, pages 110–114, 2011. (cited on Page 70)

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013. (cited on Page 6)

Christoph Molnar. *Interpretable Machine Learning*. 2 edition, 2022. URL https://christophm.github.io/interpretable-ml-book. (cited on Page 25)

Jane Morris and Graeme Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational linguistics*, 17(1):21–48, 1991. (cited on Page 14)

Pamela Munro. From parts of speech to the grammar. *Studies in Language. International Journal sponsored by the Foundation "Foundations of Language"*, 30(2): 307–349, 2006. (cited on Page 54)

Jamal Abdul Nasir, Osama Subhani Khan, and Iraklis Varlamis. Fake news detection: A hybrid CNN-RNN based deep learning approach. *International Journal of Information Management Data Insights*, 1(1):100007, 2021. (cited on Page 71)

Alexey Natekin and Alois Knoll. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21, 2013. (cited on Page 15)

Roberto Navigli. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69, 2009. (cited on Page 17)

Jacob L Nelson and Harsh Taneja. The small, disloyal fake news audience: The role of audience availability in fake news consumption. *New media & society*, 20(10): 3720–3737, 2018. (cited on Page 19)

Debora Nozza, Federico Bianchi, and Dirk Hovy. What the [mask]? making sense of language-specific bert models. *arXiv preprint arXiv:2003.02912*, 2020. (cited on Page 16)

Irina Pak and Phoey Lee Teh. Text segmentation techniques: a critical review. *Innovative Computing, Optimization and Its Applications*, pages 167–181, 2018. (cited on Page 14)

James W Pennebaker, Martha E Francis, and Roger J Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001, 2001. (cited on Page 23)

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. (cited on Page 6)

Verónica Pérez-Rosas and Rada Mihalcea. Cross-cultural deception detection. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 440–445, 2014. (cited on Page 23)

Verónica Pérez-Rosas, Bennett Kleinberg, Alexandra Lefevre, and Rada Mihalcea. Automatic detection of fake news. *arXiv preprint arXiv:1708.07104*, 2017. (cited on Page 21)

Dina Pisarevskaya. Rhetorical structure theory as a feature for deception detection in news reports in the russian language. In *Komp'juternaja Lingvistika i Intellektual'nye Tehnologii*, pages 191–200, 2017. (cited on Page 23)

Joël Plisson, Nada Lavrac, Dunja Mladenic, et al. A rule based approach to word lemmatization. In *Proceedings of IS*, volume 3, pages 83–86, 2004. (cited on Page 6)

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*, 2017. (cited on Page 23)

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018. (cited on Page 17)

Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003. (cited on Page 6)

William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971. (cited on Page 13)

Sebastian Raschka. Naive bayes and text classification i - introduction and theory, 2014. URL https://arxiv.org/abs/1410.5329. (cited on Page 15)

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019. URL https://arxiv.org/abs/1908.10084. (cited on Page 7)

Julio CS Reis, André Correia, Fabrício Murai, Adriano Veloso, and Fabrício Benevenuto. Explainable machine learning for fake news detection. In *Proceedings of the 10th ACM conference on web science*, pages 17–26, 2019. (cited on Page 25)

Martin Riedl and Chris Biemann. How text segmentation algorithms gain from topic models. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 553–557, 2012. (cited on Page 14)

Álvaro Ibrain Rodríguez and Lara Lloret Iglesias. Fake news detection using deep learning. *arXiv preprint arXiv:1910.03496*, 2019. (cited on Page 77)

Lior Rokach and Oded Maimon. Decision trees. In *Data mining and knowledge discovery handbook*, pages 165–192. Springer, 2005. (cited on Page 15)

Bernardino Romera-Paredes and Philip Torr. An embarrassingly simple approach to zero-shot learning. In *International conference on machine learning*, pages 2152–2161. PMLR, 2015. (cited on Page 17)

Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL https://www.sciencedirect.com/science/article/pii/0377042787901257. (cited on Page 13)

Sam Rowlands. Misinformation on abortion. *The European Journal of Contraception & Reproductive Health Care*, 16(4):233–240, 2011. (cited on Page 20)

Victoria L Rubin, Niall Conroy, Yimin Chen, and Sarah Cornwell. Fake news or truth? using satirical cues to detect potentially misleading news. In *Proceedings of the second workshop on computational approaches to deception detection*, pages 7–17, 2016. (cited on Page 22)

Natali Ruchansky, Sungyong Seo, and Yan Liu. Csi: A hybrid deep model for fake news detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 797–806, 2017. (cited on Page 24)

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. (cited on Page 15)

Yvan Saeys, Thomas Abeel, and Yves Van de Peer. Robust feature selection using ensemble feature selection techniques. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 313–325. Springer, 2008. (cited on Page 15)

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. URL https://arxiv.org/abs/1910.01108. (cited on Page 17)

Robert E Schapire. The boosting approach to machine learning: An overview. *Nonlinear estimation and classification*, pages 149–171, 2003. (cited on Page 15)

Robert E Schapire. Explaining adaboost. In *Empirical inference*, pages 37–52. Springer, 2013. (cited on Page 15)

Bernhard Scholkopf, Kah-Kay Sung, Christopher JC Burges, Federico Girosi, Partha Niyogi, Tomaso Poggio, and Vladimir Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE transactions on Signal Processing*, 45(11):2758–2765, 1997. (cited on Page 10)

EK Seltzer, E Horst-Martz, M Lu, and Raina Martha Merchant. Public sentiment and discourse about zika virus on instagram. *Public Health*, 150:170–175, 2017. (cited on Page 20)

Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Alessandro Flammini, and Filippo Menczer. The spread of fake news by social bots. *arXiv preprint arXiv:1707.07592*, 96:104, 2017. (cited on Page 2)

Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation functions in neural networks. *towards data science*, 6(12):310–316, 2017. (cited on Page 7)

Baoxu Shi and Tim Weninger. Discriminative predicate path mining for fact checking in knowledge graphs. *Knowledge-based systems*, 104:123–133, 2016. (cited on Page 23)

Wei Shi and Vera Demberg. Next sentence prediction helps implicit discourse relation classification within and across domains. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 5790–5796, 2019. (cited on Page 16)

Maniz Shrestha. Detecting fake news with sentiment analysis and network metadata. *Earlham College, Fall*, 2018. (cited on Page 77)

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter*, 19(1):22–36, 2017. (cited on Page 22 and 24)

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. defend: Explainable fake news detection. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 395–405, 2019. (cited on Page 25)

Michael Siering, Jascha-Alexander Koch, and Amit V Deokar. Detecting fraudulent behavior on crowdfunding platforms: The role of linguistic and content-based cues in static and dynamic contexts. *Journal of Management Information Systems*, 33 (2):421–455, 2016. (cited on Page 23)

Carson Sievert and Kenneth Shirley. LDAvis: A method for visualizing and interpreting topics. In *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, pages 63–70, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-3110. URL https://aclanthology.org/W14-3110. (cited on Page 33)

Amila Silva, Yi Han, Ling Luo, Shanika Karunasekera, and Christopher Leckie. Propagation2vec: Embedding partial propagation networks for explainable fake news early detection. *Information Processing & Management*, 58(5):102618, 2021. (cited on Page 25)

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *NAACL-HLT*, 2018. (cited on Page 23)

Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):1–12, 2019. (cited on Page 12)

Trieu H. Trinh and Quoc V. Le. A simple method for commonsense reasoning, 2018. URL https://arxiv.org/abs/1806.02847. (cited on Page 7)

Udo Undeutsch. Beurteilung der glaubhaftigkeit von aussagen. *Handbuch der psychologie*, 11:26–181, 1967. (cited on Page 23)

Christian Unkelbach, Alex Koch, Rita R Silva, and Teresa Garcia-Marques. Truth by repetition: Explanations and implications. *Current Directions in Psychological Science*, 28(3):247–253, 2019. (cited on Page 56)

Sander van Der Linden, Jon Roozenbeek, and Josh Compton. Inoculating against fake news about covid-19. *Frontiers in psychology*, 11:566790, 2020. (cited on Page 20)

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. (cited on Page 7)

Cédric Villani. *The Wasserstein distances*, pages 93–111. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-71050-9. doi: 10.1007/978-3-540-71050-9_6. URL https://doi.org/10.1007/978-3-540-71050-9_6. (cited on Page 8)

Giulia Vilone and Luca Longo. Explainable artificial intelligence: a systematic review. *arXiv preprint arXiv:2006.00093*, 2020. (cited on Page 25)

Andreas Vlachos and Sebastian 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, 2014. (cited on Page 21)

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17 (4):395–416, 2007. (cited on Page 11)

Soroush Vosoughi, Deb Roy, and Sinan Aral. The spread of true and false news online. *science*, 359(6380):1146–1151, 2018. (cited on Page 20)

Atro Voutilainen. *Part-of-speech tagging*, volume 219. The Oxford handbook of computational linguistics, 2003. (cited on Page 6)

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018. (cited on Page 16)

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020. (cited on Page 7)

Jonathan J Webster and Chunyu Kit. Tokenization as the initial phase in NLP. In *COLING 1992 volume 4: The 14th international conference on computational linguistics*, 1992. (cited on Page 6)

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016. (cited on Page 17)

Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001. (cited on Page 12)

You Wu, Pankaj K Agarwal, Chengkai Li, Jun Yang, and Cong Yu. Toward computational fact-checking. *Proceedings of the VLDB Endowment*, 7(7):589–600, 2014. (cited on Page 23)

Fan Yang, Shiva K Pentyala, Sina Mohseni, Mengnan Du, Hao Yuan, Rhema Linder, Eric D Ragan, Shuiwang Ji, and Xia Hu. Xfake: Explainable fake news detector with visualizations. In *The World Wide Web Conference*, pages 3600–3604, 2019. (cited on Page 25)

Wenpeng Yin, Jamaal Hay, and Dan Roth. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach, 2019. URL https://arxiv.org/abs/1909.00161. (cited on Page 52)

Mikhail Yurochkin, Sebastian Claici, Edward Chien, Farzaneh Mirzazadeh, and Justin Solomon. Hierarchical optimal transport for document representation, 2019. URL https://arxiv.org/abs/1906.10827. (cited on Page 7)

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1476–1485, 2019. (cited on Page 17)

Xueyao Zhang, Juan Cao, Xirong Li, Qiang Sheng, Lei Zhong, and Kai Shu. Mining dual emotion for fake news detection. In *Proceedings of the Web Conference 2021*, pages 3465–3476, 2021. (cited on Page 21)

Lina Zhou, Judee K Burgoon, Jay F Nunamaker, and Doug Twitchell. Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated communications. *Group decision and negotiation*, 13(1):81–106, 2004. (cited on Page 23)

Xinyi Zhou and Reza Zafarani. Fake news: A survey of research, detection methods, and opportunities. *arXiv preprint arXiv:1812.00315*, 2, 2018. (cited on Page 23)

Xinyi Zhou, Reza Zafarani, Kai Shu, and Huan Liu. Fake news: Fundamental theories, detection strategies and challenges. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 836–837, 2019. (cited on Page 25)

Xinyi Zhou, Atishay Jain, Vir V Phoha, and Reza Zafarani. Fake news early detection: A theory-driven model. *Digital Threats: Research and Practice*, 1(2):1–25, 2020. (cited on Page 21)

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books, 2015. URL https://arxiv.org/abs/1506.06724. (cited on Page 7)

Gregory D Zimet, Zeev Rosberger, William A Fisher, Samara Perez, and Nathan W Stupiansky. Beliefs, behaviors and hpv vaccine: correcting the myths and the misinformation. *Preventive medicine*, 57(5):414–418, 2013. (cited on Page 20)

Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one*, 11(3):e0150989, 2016. (cited on Page 21)

Chaoyuan Zuo, Ayla Karakas, and Ritwik Banerjee. A hybrid recognition system for check-worthy claims using heuristics and supervised learning. In *CEUR workshop proceedings*, volume 2125, 2018. (cited on Page 21)

I herewith assure that I wrote the present thesis independently, that the thesis has not been partially or fully submitted as graded academic work and that I have used no other means than the ones indicated. I have indicated all parts of the work in which sources are used according to their wording or to their meaning.

Magdeburg, 26th July 2022