

An investigation of Computational Propaganda Detection via NLP

CS310

Alexey Savelyev

Supervisor: Dr. James Archibald
Department of Computer Science
University of Warwick
2024-2025

Abstract

Propaganda is a prevalent tool used for, among many other things, the manufacture of public consent, misinformation, and the spread of radical ideology. Detection is more important than ever yet most detectors underperform in real-life settings. This project explores the task of Computational Propaganda Detection providing a brief survey paper on the field and adjacent relevant topics. Various representations, types and features of propaganda are discussed. Based off the initial research the most promising avenue of propaganda detection is explored, which is detection of persuasive techniques. Architectures for span identification and technique classification of these techniques for journalistic text are developed and evaluated, based off known best-performing architectures. Some additional features are experimented with. The findings suggest that fine-tuned custom models outperform pre-trained encoder models (RoBERTa) by a thin margin, while significantly outperforming Large Language Models. Future research directions for propaganda detection are proposed, which primarily revolve around refining the definition of propaganda in a computational setting.

Keywords: Machine Learning, Artificial Intelligence, Natural Language Processing, LSTM (Long Short Term Memory), Large Language Models (LLMs).

Acknowledgements

I would like to thank Dr James Archibald, who has continually supported me throughout the development of this project. I would also like thank Dr Maryam Mousavi who also provided consistent guidance throughout the project - this project would likely not be in its current standard without her support. Finally, I would like to thank my family and close friends, who's moral support made this project possible.

Contents

1	Introduction	5
1.1	What is Propaganda?	5
1.2	Objectives	5
2	Background	6
2.1	Computational Propaganda	9
2.2	Existing detectors	9
3	Technologies	10
3.1	Natural Language Processing	10
3.1.1	Word Embeddings	10
3.1.2	Early Approaches	11
3.2	Artificial Neural Networks	11
3.2.1	Optimizers	13
3.2.2	LSTMs	15
3.2.3	Transformers	17
3.2.4	BERT	19
3.2.5	Conditional Random Fields	19
3.2.6	Large Language Models	20
3.2.7	Benchmarks	20
3.2.8	DeepSeek	20
4	Computational Propaganda Detection	22
4.1	Network Analysis	22
4.1.1	Malicious Actors	22
4.1.2	Initial Designs	23
4.1.3	Group Analysis	23
4.2	Natural Language Processing	24
4.2.1	Document Wide-Classification	24
4.2.2	Fine-grained analysis	25
4.2.3	Joint tasks	27
4.2.4	Adjacent fields	28
5	Methodology	29
5.1	Research	29
5.1.1	Research Management Tools	29
5.2	Development	29
5.3	Ethical and Legal considerations	30
6	Design and Evaluation	31
6.1	Languages and Tools	31
6.2	Preprocessing	31
6.3	Baselines	32
6.3.1	Large Language Models	32
6.4	Custom architectures	33
6.4.1	Span Identification	33
6.5	Technique Classification	36
6.6	Considered approaches	37
7	Conclusion	40
7.1	Project Management	40
7.2	Literature review and future work	40
7.3	Development	41
7.4	Self-reflection	41
7.5	Final thoughts	41

1 Introduction

1.1 What is Propaganda?

The word *propaganda* can be derived from the Latin *propagare* - to spread or propagate. Originally, it referred to the spread of Catholic faith in the 17th century ([47], p2). Propaganda as a cultural phenomenon has existed long before its definition, from ancient Assyrian slanderous war poems used to demoralise their opponents, to mystified stories of kings used to proclaim their divine right to rule [86]. Since then its taken on a wide range of meanings [47], usually dependent on which discipline it is being discussed by; sociology, political science, psychology, or through interdisciplinary pursuits. One popular definition is as follows[39] - '*expression of opinion or action by individuals or groups deliberately designed to influence opinions or actions of other individuals or groups with reference to predetermined ends*'. Traditionally, propaganda is seen as a negative term, referring to the spread of highly biased misinformation sourced by governments. It is rare (but not unseen in some cultures) for a state to label any statements made by itself as anything but the absolute truth. Typically, the label of propaganda is instead assigned by the state's opposition, regardless of the validity of the statement (known as poisoning the well [72]). Despite often being state-approved, propaganda is not strictly reserved for governments [14] and has seen its use by corporations, often labelled as PR [54], with a blurry line between the two concepts which is often disputed.

Unfortunately, this definition, as do most pertaining to propaganda lacks specificity and therefore leaves room for interpretation to what propaganda actually is. Some questions are raised; is propaganda inherently malicious? Is propaganda exclusive to the political realm? In what manner does propaganda influence the individual's opinion? How does propaganda spread? Unfortunately, there are few concrete answers for these questions. In practice, propaganda is easier to identify and define through which advertise a specific view, typically sharing common strategies between campaigns.

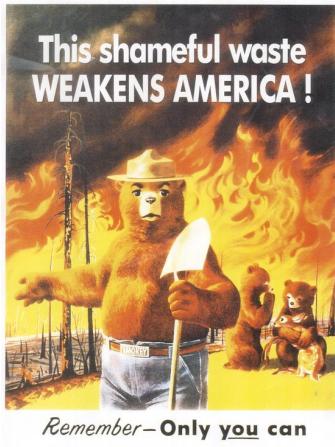
This project aims to both research the field of computational propaganda detection, develop some of the best known systems for it, and attempt to improve upon them. Given the nature of this project a wide range of topics and models were explored - some of which weren't aren't directly relevant to the models that were developed in the end. It is however important to understand why these were not chosen, and what can be learned from them. As such, this document serves in part as a log of considered attempts/approaches, with discussions as to why these were not successful.

1.2 Objectives

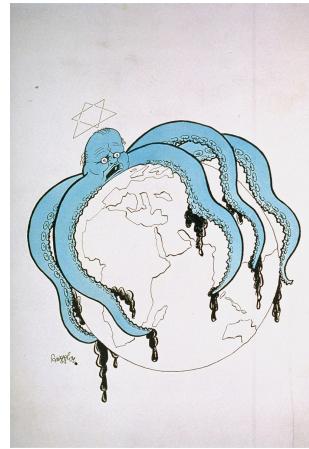
The objectives of this project are as follows;

1. Research of the various features and architectures used in propaganda detection, producing an in-depth summary paper.
2. Full implementation of a small handful of the current top-performing architectures for the task of computational propaganda detection.
3. An ablation study of the developed architectures, including the testing of various proposed new additions.

Note that the summary, or 'survey' paper this goal refers to comprises the research section of this document. At the beginning of the project, the best-performing approaches for computational propaganda detection were not clear, and therefore a large amount of time and effort was put into researching this field. Once the research done was deemed sufficient, the models were developed and evaluated. There is no specific number of models that was necessary to develop to deem the project a success - this was kept intentionally open-ended to allow the project to be scaled back, or scaled up depending on the progress being made. A similar thought process was used for the proposal of various features. Note that originally, 2 more objectives were present - a thorough evaluation of the developed models and proposal of an alternate architecture for propaganda detection. These are both however encapsulated by the third objective - an ablation study is an evaluation of the different components of an architecture, and an alternate architecture is implicitly created via any additions to it.



(a) Propaganda poster for The Smokey Bear Wildfire Prevention campaign, ran in the USA in the 1920s.



(b) Nazi propaganda cartoon by Seppla (Josef Plank) [4]. Depicts Winston Churchill as an octopus with the Star of David above his head. A widespread nazi myth existed that Jews were engaged in a conspiracy to take over the world. Exact date unknown.

Figure 1: Examples of both debatably positive and negative uses of propaganda - preventing wildfires, or in contrast, justifying war.

2 Background

Propaganda could be said to predate civilization - Neolithic cave drawings of men using weapons against one another fall into the provided definition [86]. While no written records from said time exist (nor could they), the intentional influencing of others opinions could be done through visual mediums, or any other methods of conveying meaning. It wouldn't be unreasonable to assume that propaganda is as old as the concept of communication. Ancient Greece saw more sophisticated applications of propaganda, such as utilising basic deception to avoid conflict, or the use of omens to influence the morale of soldiers. Alexander the Great's claim of being Zeus' son, placement of his face on coins, and general self-deifications can all be seen as forms of political propaganda. Looking further ahead into humanity's history, propaganda continued growing in its sophistication, with much of that being in the form of religious propaganda. An example is ecclesiastical edicts (laws) and papal bulls, which asserted doctrinal conformity and motivated crusading armies. Royal and noble patrons further commissioned lavish illuminated manuscripts and church-sponsored art like stained glass windows to communicate dynastic legitimacy and moral instruction to largely illiterate populations.

Post-nineteenth century propaganda existed in various mediums, and its use was honed in the many conflicts during the twentieth and twenty-first centuries. These include, but are not limited to: news tabloids, television, radio, film, animation, literature - any form of media that can be consumed by an individual can be propaganda. Propaganda played a key part in these conflicts - keeping morale up was vital for maintaining civilian labour and military strength, whilst demoralising the enemy population stripped them of these resources. What marks a distinction in modern propaganda, especially propaganda from the latter half of the twentieth century is the rise of globalisation - propaganda that was previously geographically constrained could spread worldwide. Nowadays, one largely popular medium is the Internet. At first glance, the Internet appears to expand political and social freedom by providing an anonymous platform for discussion and communication. Scientific literature [77] suggests that this is not the case. Authoritarian regimes tend to utilise the internet as a way of shaping internal (the subjects of the regime's) opinions, as they have done with other mainstream forms of media, with the additional benefit of monitoring potential dissenters via direct surveillance. Conversely, the Internet provides an anonymous, accessible medium for spreading propaganda to those outside the regime. These distinct approaches may be referred to as *internal* or *external* propaganda [80] i.e. propaganda for those subject to the regime, or propaganda for those outside it. The following are some examples of the usage of both *internal* and *external* propaganda by various entities.

Authoritarian Propaganda

Modern China's propaganda system is interwoven into almost all Chinese pieces of media: "newspaper offices, radio stations, television stations,..., and other cultural facilities and commemoration exhibition facilities" [80]. With a sprawling infrastructure for censorship and spread of state-approved media, Chinese propaganda is defined by its efforts for direct state media control, and consequently endorsement of

specific state-approved media. The role of modern Chinese propaganda is not as prevalent as it was during its Maoist era, which among other methods, included incarceration with the intent of brainwashing, direct control of educational curriculum, domination of broadcast media, as well as journalistic sources, and the use of propaganda teams (xuanchuan du) for indoctrination of specific populations. As of 2003, the Chinese Communist Party Department utilises a large number of channels including 2,262 television stations (98% of which are local), 2,119 newspapers, 9,074 periodicals and 1,123 publishing houses which published 190,391 books. This is not accounting for the direct policing of internet usage (yintewang) [80], or the Great Firewall [47] [37]. Propaganda (xuanchuan) is spread with the intent aim of educating the masses, in hopes of building a "socialist spiritual civilization" or "harmonious society" [80]. The concept of xuanchuan lacks the typical negative connotations of propaganda in western countries, and is as such its intent is explicitly known. This is in stark contrast to the concept of propaganda typically seen in the West with its many negative connotations.

Manufacturing Consent

Some models discuss how propaganda may exist in a democratic setting. These models claim that propaganda campaigns can still be ran effectively with freedom of the press in place. The line of thinking is as follows; unlike China, the U.S. does not explicitly censor its media sources. Instead, via the structure of the free market economy, mass media companies implicitly prioritise the interest of wealthy investors and their government. The model of most relevance to this discussion is the one proposed in *Manufacturing Consent* by Noam Chomsky and Edward Herman, which was initially proposed in 1989 [9]. The model is comprised of the 5 "filters" for which the authors claim that all mass media in the western world must go through. Although originally a model exclusively for the US propaganda system, Chomsky and Herman argue that it can be applied to any similarly structured economy. The following are the 5 filters argued for in *Manufacturing Consent*.

1. **Ownership** - This entails the concept that mass media is either output by a large corporation or by large conglomerates of companies. This is by nature of mass media - it is expensive to run a news outlet, film movies, or publish books. It is even more expensive to do so at a high quality on a mass scale. Conversely, media companies can also be highly profitable whilst being fully integrated into the free market - there is incentive for wealthy investors to buy shares. As such stories which placate wealthy investors are considered more healthy for the company, whilst those critical of them are instead discouraged. Building rapport with the government is also a convenient way of providing incentive for minimal corporate tax and the passing of favourable labour laws.
2. **Advertising** - The idea that mass media must prioritise the interest of advertisers. The argument is as follows; whilst in early days of the tabloid industry profits could be derived from the cost of the product i.e. the newspaper, ad-based outlets beat out their competitors by providing the consumer with a cheaper product. By extension, media companies who's target demographic could afford to buy the advertised goods were prioritised in regards to securing sponsorship. As such, the incentive shifted from creating media that attracted consumers, to creating media that attracted consumers with buying power. As such, media critical of large corporations, depicting news that go against the narrative, or in general too complex or serious is too risky for media companies. Firstly, they may prove to be too unsanitary or impact the consumer's desire to buy products. potentially missing out on much needed profit. Secondly, media of that nature would likely deter any potential sponsors. The creation of quality media no longer becomes the primary goal - instead, media placating advertisers is made.
3. **Sourcing** - mass media often relies on a small handful of sources for the majority of their information. It is therefore argued that this causes an implicit bias in news coverage of stories. It is simply too expensive to have reporters everywhere where important stories may occur. Typically these are critical government locations such as the White House, 10 Downing Street, etc. In effect, the process of publishing information from these sources becomes routine and journalism deviating from this becomes the far more inconvenient option. This gives the governmental body direct control of what information the media company has access to. This can be used to frame latest news coverage towards specific stories or perspective by over-representing these stories in conferences, by refusing to acknowledge certain narratives, or by overemphasising their expertise. Furthermore, Chomsky and Herman claim that often an 'expert' is chosen to confirm the desired narrative. They claim that typically they are chosen precisely because their personal opinions fit the desired narrative.
4. **Flak** - The fourth filter than Chomsky and Bennett argue for. This filter takes the form of negative responses (letters, petitions, lawsuits, etc.) to a piece of media from businesses, government officials, etc. Flak can be costly to media companies - it may actively damage its reputation resulting in loss of advertisers. Flak is reserved for the those with influence to produce. The idea of the argument is that letters from the White House are significantly more threatening than a letter from a common

man, or lengthy lawsuits are reserved for the absurdly wealthy.

5. **Fear/Anti-Terrorism** - originally *anti-communism*, this filter describes an anti-ideology of whoever the enemy of the government is perceived to be. Originally this filter referred to the Soviet Union during the Cold War [9], with it transforming into *anti-terrorism* after its collapse. Any supposed support of this entity is heavily scrutinised (even if warranted), whilst slander is encouraged regardless of its veracity. This goes beyond genuine support of the opposition - due to the vagueness of communism, or terrorism, any challenge towards the current political landscape can be branded as support for these perceived 'evils'.

Western, specifically American propaganda is studied to a much lesser extent than propaganda in authoritarian regimes. Nonetheless, the model described is widely accepted in the sphere of political science, providing it credit. It could be argued that American propaganda is studied significantly less precisely due the described blend with the free-market economy.

Public Relations

Public relations (PR) and propaganda are closely linked in the scientific literature, both referring to organised, strategic communication designed to shape public opinion. Classic PR scholarship describes the field as a "planned, deliberate and sustained effort to influence opinion through socially responsible performance and two-way communication" [22], whereas propaganda is defined as a systematic attempt to mould perceptions and direct behaviour to serve the communicator's agenda [47]. Because both definitions hinge on intentional persuasion, several authors argue that PR and propaganda rely on fundamentally similar techniques [63], and some go further, contending that PR is "propaganda by another name," differing more in connotation than in method [64]. Thus, many academics see substantial conceptual overlap between the two terms, even if their ethical reputations diverge in practice.

These 3 examples of different understandings of propaganda highlight the challenge in defining and thus classifying propaganda algorithmically. Based off existing literature, it may be reasonable to assume the use of propaganda by all political entities - maliciously or not. Propaganda may be engrained into a free-market economy making it more difficult to identify via source (*Black Propaganda*), it may be explicitly stated to be propaganda without any negative connotations (*White Propaganda*), or it may instead be labelled as 'PR'. All of these approaches aim to influence the opinion of the reader and as such aim to be as persuasive as possible, but their sources of delivery, their medium and their approach to persuasion and as such will likely have distinct properties that can be used for detection. The 3 examples described above are just some forms propaganda may take - to complicate the situation even further, propaganda may also not always aim to persuade - it may aim to intimidate. Another set of mutually exclusive categories is as follows;

1. *Soft propaganda* - propaganda which aims to persuade, relating to the concept of *soft power* [68].
2. *Hard propaganda* - propaganda which doesn't aim to sway the opinion of its consumer, it instead aims to intimidate. It is typically over-the-top, and often used as a form of *internal propaganda* to discourage political awareness. There would therefore be less value in detecting this type of propaganda - the issue doesn't lie with the information contained, but by the fact that this propaganda exists. It acts as a symbol of power of a regime over its people.

To add a further layer to the confusion, another distinction should also be made from similar but related concepts, such as *misinformation* or *satire* [76]. These may have similar characteristics, but serve different purposes. Misinformation is defined by the spread of false claims whereas propaganda contains no such certainty - the truthfulness of its content is unknown, and it often contains some of the truth (to improve its credibility). Satire often relies on over-the-top, absurd language that may also found in propaganda, however it's purpose is to parody and critique (typically the group being imitated). In practice these will share similar features such as over-the-top language characteristic of parodies, bipartisan tabloids and state propaganda. They should however not be labelled as one.

Propaganda is therefore a challenge to classify due to its many meanings and wide taxonomy - propaganda may be used by anyone, and will look slightly different each time. It alters its methods of delivery with the evolution of technology and can't be classified by its veracity. Over-the-top language would be a strong indicator if not for its use similar yet distinct categories of media with vastly different meanings, and even if such propaganda is classified, there may not be much value in doing so if it is already explicitly known as propaganda.

2.1 Computational Propaganda

Whilst it is the case that modern-day propaganda may present itself in various forms of media, a predominant issue in the 21st century is computational propaganda. As defined by Howard et al. [40], it is the spread of propaganda directly influenced by networks of social media, artificial intelligence, specialised algorithms and big data. Computational Propaganda relies on the same techniques for persuasion, but differs from other mediums due to its scale, accessibility, and ability to generate personalised propaganda for its target. This typically takes the form of targeted social media campaigns via the use of *botnets*, a semi-automatic collection of AIs used for the spread of propaganda. Additionally, due to anonymous nature of the Internet, the platform lends itself to easy use of Large Language Models (LLMs) for impersonation of real users, as well as the use of Deepfakes or other content created by generative AI. The rate of growth of these technologies has proven to be counteractive towards many potential detection methods.

2.2 Existing detectors

At the time of writing this, no widely distributed computational propaganda detection engine exists; given the numerous definitions, classifications and distinctions that exist this is unsurprising. Defining propaganda is difficult enough - identifying computationally identifiable features is even harder. Malicious actors can also alter their behaviour to avoid detection [18], making preventative measures almost always reactionary. Whilst various avenues of research have been explored (which will be shortly discussed), the performance of the models is often lacklustre, operating better in theory than in practice in part due to this reactionary nature. Nonetheless such systems could in theory provide defence to political tampering via propaganda campaigns online which have proven to be an urgent issue [10]. Identification doesn't entirely prevent these campaigns, however to persuade, a source should not seem biased - there is therefore vested interest in staying unidentified. This would also provide an avenue for better content moderation on social media platforms, which could protect themselves against being misused by *botnets*.

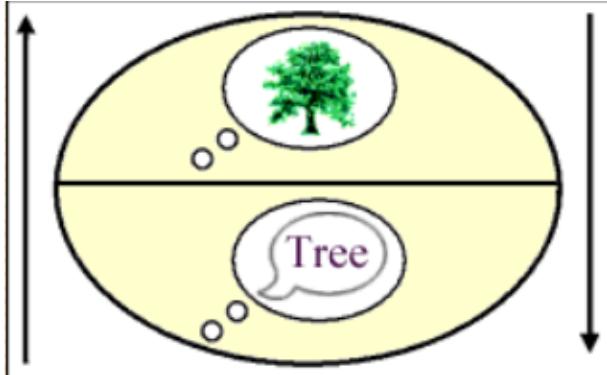


Figure 2: Example of a sign, with the signifier on the bottom, and signified on the top. Taken from Daniel Chandler [79].

3 Technologies

To choose the correct tools for computational propaganda detection they must first be understood.

3.1 Natural Language Processing

Natural Language Processing [96] is a subfield of computer science focused on a computer’s ability to interact with natural human-written language. Given such a vague definition, there are many tasks encapsulated within it, from deriving meaning via summarisation, labelling parts of speech, translation, or prediction of the next word in a sequence. Discussing the field in its entirety isn’t feasible - instead, some key concepts are mentioned instead. The Technologies section of this document is vaguely split into NLP and Neural Networks - please note that these overlap heavily and the later sections of Neural Networks is in essence a continuation of the discussion of NLP.

3.1.1 Word Embeddings

To interact with natural language, the words must first be transformed into a format that a computer may process and manipulate. From a purely text-based standpoint, this is not difficult to do - ASCII already represents characters in numerical form, and thus words can be ‘understood’ by assigning them some arbitrary value. The difficulty comes in mapping the nuance and subtlety that is fundamental within spoken language. How could a computer understand the meaning of a word? This problem is in line with *semiotics* (study of signs and signals); specifically the concept of a *signifier* and *signified*. The signifier is the label for a concept, whereas the signified is the concept itself (an example can be seen in figure 2). Much of Natural Language Processing revolves around building a better translation from signifier (word) to signified (concept).

One initial idea is the linking of words manually via similarity. One famous example is WordNet [98], a lexicon (collection of words) containing most of the English language with synonyms and hypernyms grouped into the same categories generating a large graph. Similarity of words can be compared directly using this mapping. That being said this approach is largely considered to be flawed - words can often have more than one meaning e.g. ring, bat, sink, and there is no elegant way of solving this issue via grouping of synonyms. Furthermore, new or misspelled words can’t be effectively grouped, requiring manual addition to the mapping. Counter-intuitively only relying on closely related words builds a poor understanding of them, and as such other approaches must be explored.

The meaning of a word is better mapped when it is ignored. Instead, by looking at which words are often in close proximity with the target word and are often seen together a more nuanced representation can be extracted. The intuition behind this can be summarised by a single quote; “*You Shall Know a Word by the Company It Keeps*” [45]. Words with similar meanings often find themselves surrounded by the same subset of other words e.g. “I am *tired*” vs. “I am *exhausted*”. The most popular approach for numerically storing this information is a *word embedding*. *Word embeddings* are a vectorised representation of a word and as such, belong in a latent space. Words which share the same meaning can be vectors similar to one another (via cosine similarity), or in contrast be opposites¹. Much of current research in NLP revolves around making these embeddings as robust as possible - the more semantic meaning that can be captured

¹In practice this is a large simplification of the vectorisation approach. Whilst vector similarity was what initially drove these models, more sophisticated methods generate embeddings which don’t always follow this trend of cosine similarity being the same as similarity in meaning

in the embedding, the more complex the tasks that can be done with them are. Additionally different embedding models display different curious characteristics.

3.1.2 Early Approaches

Taking a step back, statistical approaches were the foundation of early NLP, modelling text as sequences of discrete events (words or characters) and estimating probabilities directly from large corpora. The simplest and most enduring of these are n-gram language models, which assume the next word depends only on the previous $n-1$ words and rely on smoothing to handle unseen events. From the 2000s onward, neural methods such as word2vec transformed this paradigm by learning dense embeddings that capture syntactic and semantic regularities via shallow neural networks trained on massive text collections.

N-grams

A word n -gram model estimates the probability of a sentence by the product of conditional probabilities of each word given the previous $n - 1$ words. In a bigram ($n = 2$) model:

$$P(w_1, \dots, w_T) \approx \prod_{t=1}^T P(w_t | w_{t-1}). \quad (1)$$

Despite their simplicity, n-gram models suffer from data sparsity and limited context windows [8] and are thus nowadays rarely used, although an understanding of them is still important for understanding why the current approach is in the form that it is. Statistical approaches to mapping languages were popular in various tasks, including translation and language modelling. However, these were found to be crude - neural networks simply outperformed them in most tasks while requiring significantly less or no feature engineering. For computational efficiency during runtime, statistical models may still be used - however for higher accuracy and more robust understanding of data, neural networks are preferred.

Early Language Models

Word2vec is a family of models introduced by Mikolov et al. that learn one dense d -dimensional vector per word by training a shallow neural network on very large text corpora [62]. Two main architectures are used:

- *Continuous Bag-of-Words (CBOW)*: Predicts a target word from the (average) of its surrounding context vectors within a window of size m [62].
- *Skip-gram*: Uses the current word to predict each context word in a window.

By optimising for some loss function, dense vector representation of individual words are built. A somewhat unique feature of this approach is its embeddings exhibiting linear regularities. Words embeddings that are added or subtracted to one another result in other embeddings e.g.

$$\mathbf{v}(\text{king}) - \mathbf{v}(\text{man}) + \mathbf{v}(\text{woman}) \approx \mathbf{v}(\text{queen}), \quad (2)$$

reflecting the initial hypothesis that words occurring in similar contexts tend to have similar meanings. This representation is still relatively limited - there is no part of the model which allows for the capturing the context each word is used in.

Word2vec was a nominal approach in Natural Language Processing. It minimised many of the issues with the definition based approaches, as it allowed for new words to be trained for without having to assign a meaning for them. It is also a prime example of the shift from direct, hand-crafted features for each word to dense vector representations in the form of embeddings. While word2vec uses neural networks, it does so sparingly. Its successors, as well as many models for NLP tasks rely more heavily on them, thereby generating more effective dense vector embeddings.

3.2 Artificial Neural Networks

The best performing models for computational propaganda detection are typically a derivation of an **Artificial Neural Network**, a machine learning model paradigm defined by their basis on the functionality of a mammal brain [35]. They are especially effective in highly complex tasks which require non-linear processing, and can 'learn' to behave optimally for a given task when provided a method to penalise incorrect behaviour (loss function). As a result of their learning, they are often treated as *black boxes* with no clear understanding as to why a model returns the output it does (based on its parameters). They often outperform traditional machine learning approaches, at the cost of training resources, explainability, and predictability.

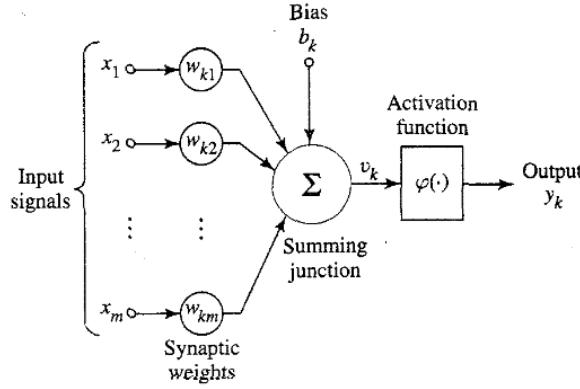


Figure 3: The structure of a single neuron unit in an Artificial Neural Network (ANN)

Particularly, they are based on the neuron, which accepts electrical signals and relays them to other neurons, allowing complex computation to occur with relatively simple single units. These 'neurons', which are placed into groups referred to as *layers*, sum all previous inputs, which are weighted by individual parameters. This is then passed through an *activation function*. The output of the activation function is then used as part of the input for the next layer of neurons. This is repeated for a set number of layers [35]. The utility of standard Feed Forward Networks has been shown to be relatively limited (in comparison to its successors) and is not suited to many tasks (such as sequential data classification). The following is the calculation for the output of a single neuron unit;

$$y_k = \varphi(b_k + \sum_{i=1}^m x_i \cdot w_{ki}) \quad (3)$$

y_k is the output of a neuron k for a given layer, x_1, x_2, \dots, x_m are the inputs to the layer, b_k is the bias value to neuron k, and $w_{k1}, w_{k2}, w_{k3}, \dots, w_{km}$ are the input-wise weights for neuron k. φ is the activation function, which is a non-linear function mapping the sum of the weighted inputs and bias value to a specific range. Note that without a non-linear function, the neural network would collapse into a linear function not making it particularly useful. By equipping the network with an activation function, non-linearity is introduced - this is the feature that allows neural networks to produce arbitrarily complex functions which allows them to be used for tasks such as propaganda detection.

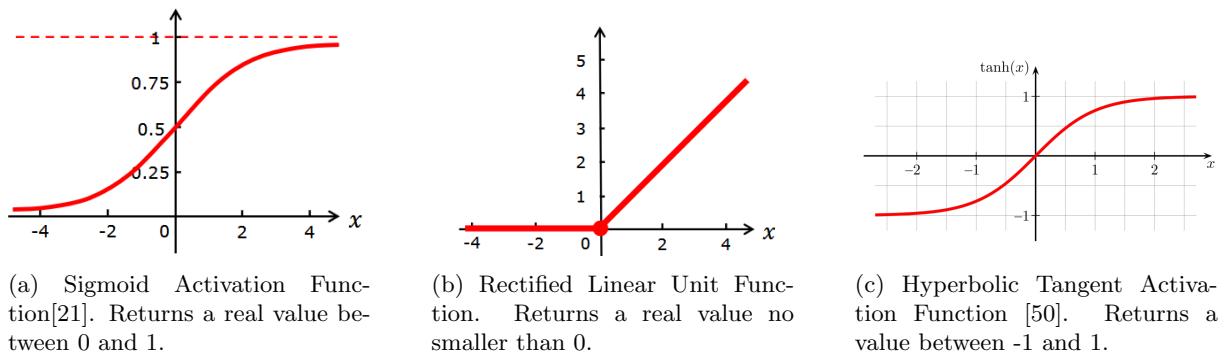


Figure 4: Relevant Activation Functions

The output is returned after the initial input is passed through the network, layer by layer, with the values stored in the neurons on the final layer being considered the output. This output is evaluated via a loss function, which models how "correct" the neural network's output is. The loss function largely depends on the task the neural network is being utilised for.

The one most relevant to the eventually developed models is *cross-entropy loss*, as it is perfectly suited for classification tasks. Its formula is as follows;

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C y_{i,c} \log(\hat{y}_{i,c}) \quad (4)$$

with \mathcal{L} being the overall cross entropy loss, N being the total number of samples (or data points) in the dataset (or a batch if computed on mini-batches) and C being the total number of classes in the classification problem. $y_{i,c}$ is the ground-truth (i.e the correct) label for sample i and class c . In many cases, these labels are one-hot encoded meaning $y_{i,c} = 1$ if sample i belongs to class c and 0 otherwise. $\hat{y}_{i,c}$ is the predicted probability that sample i belongs to class c , typically computed using a soft-max function over the model's outputs.

Softmax is key for converting raw Neural Network outputs (logits) into probabilities that can be used for concrete classifications.

$$\text{softmax}(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_{j=1}^K \exp(z_j)} \quad (5)$$

with $\mathbf{z} = (z_1, \dots, z_K)$ being the input vector of real-valued scores (logits), $\exp(z_i)$ ensuring each output is strictly positive, and the denominator $\sum_{j=1}^K \exp(z_j)$ normalizing the results so that $\sum_{i=1}^K \text{softmax}(\mathbf{z})_i = 1$. This maps the raw scores into a probability distribution over K classes, allowing us to interpret $\text{softmax}(\mathbf{z})_i = P(\text{class} = i \mid \mathbf{z})$.

Once the loss is calculated, the neural network may go through *backpropagation* - that is, the partial derivative of every trainable parameter relative to the loss function can be calculated via the chain rule. By calculating the value of the derivative, it is known whether increasing or decreasing the value of a parameter will result in a smaller loss value. As such, each parameter's value is increased or decreased to minimise the objective loss.

Propagation (forward pass of the neural network), calculation of the loss function, and back-propagation (calculating each partial derivative and changing values accordingly) is done until the loss function converges on a value, or an arbitrary limit on the number of steps is reached. This is what is known as 'training' a neural network. Typically, the entire dataset is used for training, with one iteration of the entire training data being known as an *epoch*.

The process of adjusting the model's parameters is known as *gradient descent*. This can either be done by summing the loss of all training examples (*Batch Gradient Descent*) at every single iteration, using one example at a time (*Stochastic Gradient Descent*), or by calculating the loss for a small subset of the data (*Mini-Batch Gradient Descent*). Note that when looking for a minimum of the loss function, the problem is tackled from the perspective of finding a *local minima*, that is, one of the minimum points (i.e. dips) in the derivative of the loss function. The general formula for gradient descent is as follows.

$$\theta^{(t+1)} = \theta^{(t)} - \eta \nabla_{\theta} J(\theta^{(t)}) \quad (6)$$

$\theta^{(t)}$ is the set of current parameters (or weights) of the model at iteration t . $\theta^{(t+1)}$ is the set of updated parameters after iteration t . η is the learning rate, a hyper-parameter (adjustable parameter that isn't trained) that determines the size of the steps taken during each update. $\nabla_{\theta} J(\theta^{(t)})$ is the gradient of the loss function J with respect to the parameters θ at iteration t . This represents the direction and magnitude of the steepest increase in the loss. $J(\theta)$ is the cost (or loss) function that quantifies the error between the model's predictions and the actual data.

3.2.1 Optimizers

Gradient descent is vital for the learning process of Neural Networks. However other more efficient algorithms exist. Two relevant algorithms are Adam (*Adaptive Moment Estimation*) [49] and AdamW (Adam with weight decay) [55]. Both are adaptive gradient descent algorithms - they differ from the previous methods by having a variable learning rate for each parameter.

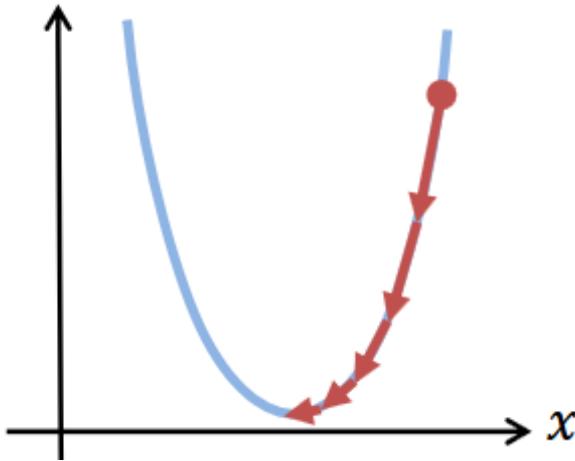


Figure 5: A visualisation of the gradient descent algorithm relative to one parameter, x . Note that in practice, this algorithm is ran to minimise the loss function (y-axis) over all parameters.

Adam computes adaptive learning rates for each parameter by maintaining two exponential moving averages: one for the gradients (first moment) and one for the squared gradients (second moment). The update rules are as follows:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \quad (7)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2, \quad (8)$$

With g_t being the gradient of the loss function with respect to the parameters at time step t , m_t being the exponentially decaying average of past gradients (the first moment), v_t being the exponentially decaying average of past squared gradients (the second moment), β_1 and β_2 are decay rates, with these dictating how much the learning rate goes down by for each iteration.

Due to initialization bias in the estimates of m_t and v_t , bias-corrected estimates are computed as:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}. \quad (9)$$

Finally, the parameters θ are updated using:

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \quad (10)$$

With θ_t representing the model parameters at iteration t , η being the learning rate, and ϵ being a small number to prevent division by zero (typically 10^{-8}).

Adam is typically chosen due to its faster convergence rate. While Adam works efficiently, it is not effective to use with *L2 regularization* a machine learning technique commonly used to prevent over-fitting (similarly to weight decay). It operates by providing a penalty term of the squared sum of weights in the loss function. In Stochastic Gradient Descent, weight decay and ℓ_2 regularization can be made equivalent. This is however not the case for Adam, and as such, it is not considered to be effective [55]. AdamW [55] decouples the weight decay term from the gradient-based update, leading to better generalization in many cases. In standard Adam, weight decay is implemented by adding a penalty (β_1) term to the loss function, which in turn affects the gradients. However, in AdamW, the weight decay is applied directly to the parameters after the adaptive update. The update equation for AdamW becomes:

$$\theta_{t+1} = \theta_t - \eta \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda \theta_t \right), \quad (11)$$

where λ is the weight decay coefficient that controls the extent of regularization.

Decoupling weight decay in this manner helps in maintaining the adaptive learning while ensuring that weight decay is effective which helps the prevention of overfitting.

Examples of sequence data		
Speech recognition		→ "The quick brown fox jumped over the lazy dog."
Music generation		→
Sentiment classification	"There is nothing to like in this movie."	→
DNA sequence analysis	AGCCCTGTGAGGAAC TAG	→ AGCCCCTGTGAGGAAC TAG
Machine translation	Voulez-vous chanter avec moi?	→ Do you want to sing with me?
Video activity recognition		→ Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	→ Yesterday, Harry Potter met Hermione Granger . Andrew Ng

Figure 6: Examples of Sequential data, sourced from AIML.com [2]

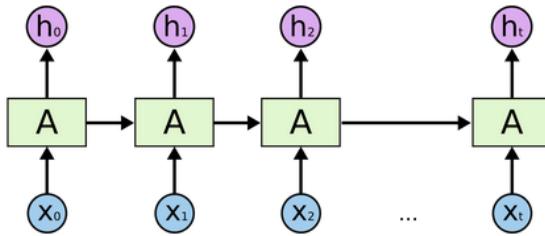


Figure 7: A basic unrolled RNN, taken from 'Understanding Neural Networks'. [89]

3.2.2 LSTMs

Neural Networks are powerful, but limited in their utility for certain classes of problems; for instance, sequential data any continuous data in which the order is relevant (some examples are shown in figure 6). This is due to the fact that traditional, or feed-forward networks accept an input of a fixed size. This is not guaranteed to be the case with sequential data. Additionally, in the standard neural architecture there is no inherent manner of mapping dependencies between ordered inputs - if you pass in words of a sentence back-to-back, there is no way of creating any temporal connection between them.

As such, different neural architectures were made to account for these problems. This class of neural networks is referred to as *Recurrent Neural Networks* (RNNs), which take in the state for a previous input to output the result of the current state, making them particularly suited for natural language processing, and by extension, propaganda detection. Unfortunately, Recurrent Neural Networks aren't perfect at handling long term spacial dependencies. As an example, in the sentence "The fool doth think he is wise, but the wise man knows himself to be a fool.", the RNN has no simple way of handling the dependencies between the word 'fool', 'man', and 'he' as they are separated by other words. This doesn't mean that these dependencies can't be learned - however, with longer sequences connections between words are lost due to the *vanishing gradient problem*. The vanishing gradient problem entails the successive nature of backpropagation - gradients for parameters are calculated based off the loss function on the final layer - if final gradients are small, the final derivative of a parameter for the beginning of the sequence may be too small to effectively train. As such the parameters responsible for dealing with inputs early on in the sequence aren't updated appropriately. The same issue occurs when attempting to learn a dependency between two connected words which are far away from one another. To combat this issue, an improvement to the RNN was proposed - The LSTM.

Similarly to RNNs, LSTMs are recurrent - they take in their prior state as an input. In addition to this however, there is a memory cell which stores information from prior inputs. It may be *outputted* or *inputted* to, or *forgotten*. This memory cell is passed into the state of the LSTM as an additional input as can be seen in figure 8; each time-step of the LSTM takes both the previous state, and the contents of the memory cell into consideration alongside the input. How this information is used, and what is stored in the memory cell is dictated via *gates*, which are in turn learnable parameters. The overall equation

for the cell state is this;

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (12)$$

Where C_t is the memory cell state at time-step t , f_t is the forget gate, C_{t-1} is the memory cell state at the previous time-step, and \tilde{C}_t is the candidate cell state. Note the use of Hadamard Product \odot [46], an element-wise multiplication of two vectors/matrices.

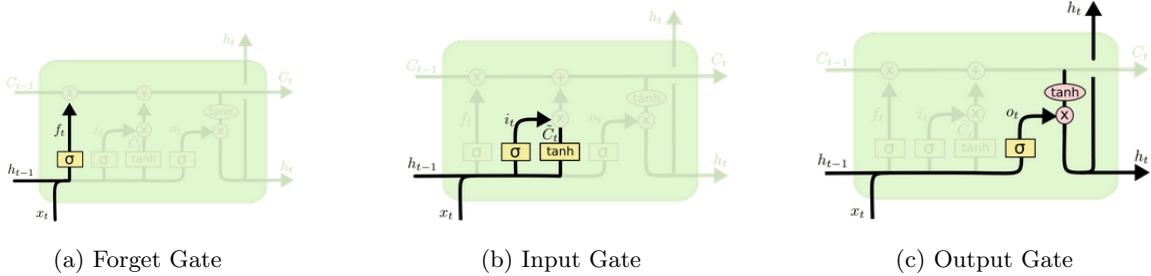


Figure 8: Visualisations of how standard LSTM gates operate.

Forget Gate

The forget gate (figure 13) controls which the previous data in the memory cell state is erased. A visualisation of the forget operation is shown in figure 8a. The calculation for this is as such:

$$f_t = \sigma(W_{fh} h_{t-1} + W_{fx} x_t + b_f) \quad (13)$$

Where f_t is a vector of values ranging from 0,1 dictating how much entry in Cell C_{t-1} is forgotten. W_{fh} and W_{fx} represent the learnable matrices responsible for the previous state and input respectively. h_{t-1} and x_t are the hidden state and input at time-step $t - 1$ and t , respectively. b_f is a bias vector. This is passed through a sigmoid ($\sigma()$)function.

Input Gate

The input gate (figure 14) is responsible for what is written to the memory cell.

$$i_t = \sigma(W_{ih} h_{t-1} + W_{ix} x_t + b_i) \quad (14)$$

Where i_t is the input-gate activation at time step t , a vector with entries in $(0, 1)$. W_{ih} and W_{ix} are weight matrices for the previous hidden state and input, respectively. i_t acts as a filter for which parts of the candidate cell, \tilde{C}_t are passed through.

$$\tilde{C}_t = \tanh(W_{ch} h_{t-1} + W_{cx} x_t + b_c) \quad (15)$$

\tilde{C}_t is the new proposed memory cell, which, like the previous gates, is based off the previous state and current input with trainable weights and bias. Unlike the gates, which act as filters, the contents of the memory cell range from -1 to 1 due to the hyperbolic tangent (tanh) function for the purpose of normalisation. Unlike the sigmoid function, tanh allows both negative and positive outputs, allowing for negative values in the cell.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (16)$$

Where C_t is the new memory cell, f_t is the forget gate, and i_t is the input gate. Certain values in the previous cell state decrease (dependent on the forget gate),the proposed memory cell is weighted according to the input gate, and their contents are summed.

Output Gate

The output gate (figure 17) determines which contents of the cell influence the output.

$$o_t = \sigma(W_{oh} h_{t-1} + W_{ox} x_t + b_o) \quad (17)$$

$$h_t = o_t * \tanh(C_t) \quad (18)$$

Where o_t is the output gate, W_{oh} and W_{ox} are the trainable weight matrices for the hidden state and input at time t.

These gates allow for more distant spacial dependencies to be learned between words. Any data saved to memory can be updated, extracted, or discarded if it is beneficial to do so and as such LSTMs are computationally powerful, and are used for a wide array of tasks. Essentially, whenever an RNN is appropriate i.e. sequential tasks, an LSTM may be used instead with potentially improved performance. However, there are some issues the model runs into. While deriving connections between inputs of spread apart time-steps is easier for LSTMs, it is not a perfect system, and data spread wide apart is hard to connect semantically. Additionally, LSTMs are computationally expensive to run - as they are sequential by nature, they are difficult to parallelize. As such, while useful, other alternatives should also be considered. Often LSTMs are stacked on top of one another, with one LSTM feeding its output into another as an input, with this potentially improving performance but making the model more difficult to train. Alternatively, LSTMs can be run independently with them taking in the input sequence in reverse order, and concatenating their output for each timestep - this is known as a *BiLSTM*. This allows for a notion of encoding information from both directions in a sequence, although this is considered a somewhat shallow approach.

3.2.3 Transformers

Transformers are a *deep-learning* architecture primarily based off the *attention* mechanism, where the encoding of a token is derived from a surrounding context window. Unlike RNNs, there is no direct notion of order - all of the words in the context window are used to make the representation for the target word equally. This model was introduced in the now nominal 2017 paper by Google, "Attention is All you need" [91]. Transformers are *sequence to sequence* (seq2seq), meaning they are comprised of an encoder and decoder component. The encoder builds a robust contextual representation of the inputted word tokens, while the decoder takes in these word tokens and returns some output. Most Large Language Models (LLMs) are built on the transformer architecture, however, most don't utilise both the encoder and decoder units. Some LLMs, such as ChatGPT, where the aim is to answer user-prompted questions, only use the decoder unit. Others, such as BERT use the encoder unit to generate better quality embeddings of words. They are also *semi-supervised*, undergoing unsupervised learning first then being finetuned in a supervised manner.

Self Attention

Attention can be thought of as a 'fuzzy' table lookup, with a query vector q , key vector k and value vector v for every single word embedding in the input. It is fuzzy in the sense that there will be more than one value returned, with all values in the table having a similarity index of between 0 and 1. This allows the forming of direct connections between words relevant to one another. This is illustrated in figure 10 - words are converted to their embedding, and for each word, individual query, key, and value vectors exist which are learnable parameters. The dot product of these vectors is calculated, then normalised via a soft-max calculation and division by the square root of the dimension size of the key vector. This 'similarity' score is then multiplied by each respective v of all words in the input. These products are summed to generate a word representation based off all of the other inputted words. As mentioned before, there is no direct notion of order - as such spacial distance between each word no longer poses an issue.

Multi-headed attention expands on this by providing multiple attention *heads* i.e multiple instances of the vectors q , k , and v . This allows for each word to be represented via its context in multiple different manners - the intuition behind this is that the other words may be relevant to this one in different ways e.g. names for the same entity vs grammar dependencies. Internally these are defined as separate Q (query), K (key) and V (value) matrices. These all undergo the same transformer process, are concatenated, multiplied by a matrix W_o to obtain a final representation. This self attention mechanism is typically repeated for a set number of times before outputting the final embedding.

While a direct notion of order has been removed, minimising the vanishing gradient problem, order still matters for a sentence. As such, encodings of each word's position are added to the calculation - these are concatenated to the initial word embedding. The positional encoding is calculated as such:

$$\text{PE}(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \quad (19)$$

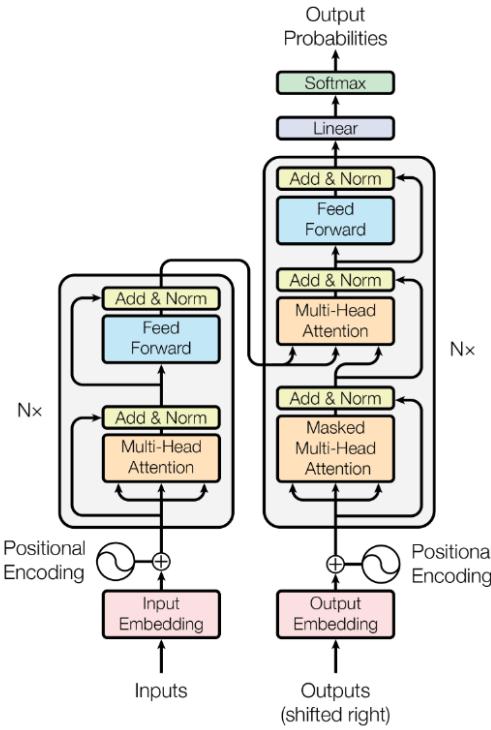


Figure 9: A standard transformer encoder decoder model.

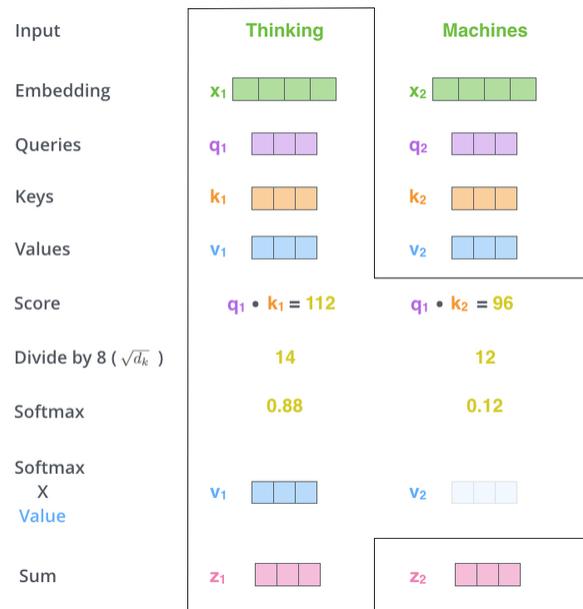


Figure 10: Example of the attention mechanism for a transformer, taken from [3].

$$\text{PE}(pos, 2i + 1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \quad (20)$$

With d being the dimension size (i.e vector length) of the positional encodings, and pos the position of the word in the input sentence.

In Figure 9, some *residual* connections can be seen - that is, a shorter path that bypasses some layers of the model. After multi-head attention, the input embeddings are summed with the self-attention word representations and normalised [5]. This is then fed through a feed-forward neural network, then summed with itself again and normalised.

So far the process described has been through the perspective of the encoder section of the model i.e. creating the dense embedding for each word. The decoder model takes in the context-rich representations of word made by the encoder, and then uses them for some predefined task. These representations are passed through another multi-head attention mechanism, FFN, and normalisation step for a number of times, then through a linear feed forward network and soft-max step. Previous outputs of the decoder are also passed into itself, allowing it to take in context from its full response. Note that the multi-head attention mechanism for outputs is masked - the decoder model cannot represent words with embeddings of words after it. Without this, the model would be allowed to effectively 'cheat' for language modelling and question answering tasks by knowing the correct answer i.e. the word next in the sequence. As such, *most* transformer based models are trained left to right, which limits the amount of context each embedding receives - a limitation that is tackled by a specific transformer based architecture.

3.2.4 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a transformer-based model that differs from previous left-to-right or right-to-left language models by jointly conditioning on both left and right context in all layers, thereby learning truly bidirectional representations [28]. Where traditional models predict the next token in sequence, BERT employs a Masked Language Modeling objective: during pre-training, a fixed percentage (typically 15%) of input tokens are replaced with a special [MASK] token, and the model must predict the original token from its bidirectional context. This enables richer contextualization since each layer's self-attention heads can attend to tokens on both sides of the mask [28].

In addition to Masked Language Modelling , BERT introduces a Next Sentence Prediction (NSP) task to capture inter-sentence coherence: given sentence pairs (A,B)(A,B), the model must classify whether BB truly follows AA in the original text or is drawn randomly from the corpus. Next Sentence Prediction helps BERT learn relationships across sentence boundaries, which proves beneficial for downstream tasks such as Question Answering and Natural Language Inference [28], although in practice, this task is not utilised for both the experiments in this project and future iterations of BERT (RoBERTa).

The base version of BERT stacks 12 transformer encoder layers, each with 12 attention heads and a hidden size of 768. BERT is trained on a large corpus of data, making it a strong baseline Lanuage Model - it is however most suited towards specific classification tasks after some *fine-tuning*. During *fine-tuning*, task-specific input and output layers are added on top of the pre-trained encoder, and the entire network is trained end-to-end on labelled data with minimal architectural changes. While the entire network is trained, due to the fact that the BERT component of the model has already been trained for language modelling (or some other task depending on the specific BERT version) the model typically performs very well even with minimal training. This concept is known as *transfer learning*, where knowledge gained from one task is reused to improve performance on a related task. In practice, since its creation, other versions of BERT which improve upon it in some manner have been developed - *RoBERTa* improves its performance, and *DistilBERT* and *ALBERT* are computationally cheaper. Nonetheless both *BERT* and its family of models are fantastic encoders and as such are a strong choice for Computational Propaganda Detection.

3.2.5 Conditional Random Fields

Conditional Random Fields (CRFs) [85] are sequence-labelling models that learn the probability of an entire tag sequence given the input sequence, $p(\mathbf{y} | \mathbf{x})$, so they avoid the strong independence assumptions made by generative models like Hidden Markov Models. In practice these help map strict dependencies between ordered labels and as such are particularly helpful when adhering to a labelling format with set transition rules. . Modern NLP systems often place a small CRF layer on top of contextual encoders (e.g. LSTM or BERT) so the model can enforce valid transitions between tags; the LSTM-CRF architecture of Huang et al. (2015) remains a popular off-the-shelf baseline [41].

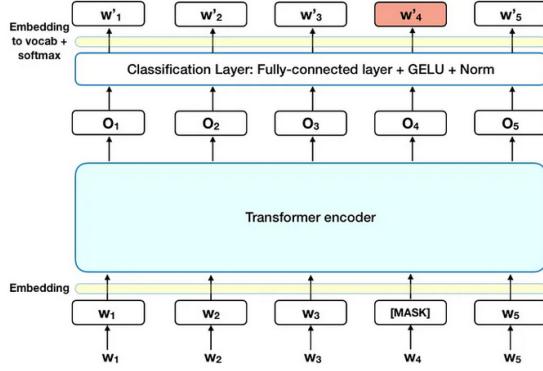


Figure 11: Schematic of BERT’s bidirectional Transformer encoder, showing masked tokens ([MASK]) and the NSP classification head.

3.2.6 Large Language Models

Large Language Models (LLMs) are neural networks trained to perform language modelling, i.e., assigning probabilities to word sequences by optimizing the next-token prediction objective [8, 74].

$$\mathcal{L} = - \sum_{t=1}^T \log P(w_t | w_1, \dots, w_{t-1}) \quad (21)$$

Training for this task can lead to models for other more useful such as autocompletion, machine translation, question answering, and summarization via prompt-based interaction [74, 75]. They are deemed “large” owing to parameter counts that have ballooned from millions to hundreds of billions—GPT-3 with 175 B parameters being a classic example [11]. Architecturally, virtually all modern LLMs derive from the Transformer, which as discussed previously, abstains from recurrence and convolution in favour of multi-headed self-attention and position-wise feed-forward layers making it much easier to train, which is what allows for these exceptionally large parameter numbers to be feasible to learn. Since 2020, the field has witnessed an explosion of scale and capability—from GPT-3 to PaLM’s 540 B parameters [17] most recently resulting in emergent reasoning. These models now routinely automate tasks once thought to exclusively possible by humans, generating coherent prose and synthesizing code with minimal supervision [74, 75]. A remarkable benefit of scale is *zero-* and *few-shot learning*, where LLMs perform new tasks from prompts alone without any fine-tuning [11]. However, they are exceptionally expensive to run on local machines (again, in part due to the number of parameters) and as such their practical usage relies on APIs or client libraries. This is an issue for the given task of propaganda detection - these models often have server-side internal limits on what they will respond to for the sake of user safety or adhering to company values. This can however result in censorship which would prevent the model from being able to operate for computational propaganda detection. Consider Figure 12 - Deepseek R1, a state-of-the-art Language Model refuses to process phrases related to blacklisted topics, such as the Tienanmen Square 1989 Protests [16]. A system such as a propaganda detection tool must not have such biases, which makes the use of LLMs for this task an option to be carefully explored and critiqued.

3.2.7 Benchmarks

Despite their excellence in zero-shot settings, LLMs exhibit varying strengths depending on training data and architectural choices, necessitating standardized benchmarks to quantify their for their various use-cases. Core linguistic benchmarks include GLUE—covering tasks such as sentiment analysis, paraphrase detection, and natural language inference—and its more challenging successor SuperGLUE, which adds Boolean question answering (BoolQ), commitment bank (CB), and word-in-context (WiC) tasks, among others [92, 93]. These are the most relevant benchmarks for computational propaganda detection - while many others exist, they check other aspects of LLMs, such as their mathematical skills or programming capabilities. Different LLMs are suited to different tasks.

3.2.8 DeepSeek

DeepSeek-R1 is an open-source Mixture-of-Experts (MoE) language model released under the MIT licence by Hangzhou DeepSeek AI in January 2025[27]. By activating only a sparse subset of its 67 billion parameters for each token, the model keeps inference costs close to those of 13 billion-parameter dense

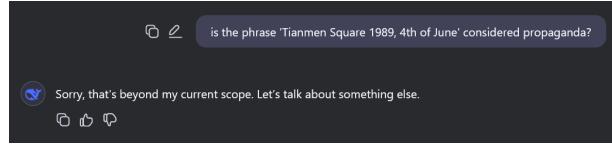


Figure 12: Deepseek refusing discussion of sensitive topics in relation to the Chinese Government

LLMs while retaining high capacity. After pre-training on ≈ 14.8 trillion tokens, the developers replaced supervised fine-tuning with Group Relative Policy Optimisation (GRPO)—a pure reinforcement-learning scheme that rewards correct chain-of-thought answers—yielding sizeable gains in mathematical and coding tasks. Public benchmarks show that R1 is comparable to OpenAI’s o1 model while being significantly cheaper to train. It is particularly suited for the given task due to its high scoring in GLUE/SuperGLUE while being available for local hosting which can be used to bypass server-side response censorship,

4 Computational Propaganda Detection

Computational Propaganda Detection is an up-and-coming field, which overlaps with various other fields in Computer Science. Specifically, it overlaps heavily with *Network Analysis* and *Natural Language Processing* [57].

4.1 Network Analysis

Computational propaganda is often propagated in Online Social Networks (OSNs). With 5.4 billion people using some sort of social media on a regular basis [43], propaganda spread through these networks has a massive target audience. Furthermore, many platforms have an inbuilt curation function, showing users content that would interest them most. By pandering to specific target demographics, such as groups susceptible to misinformation, the propaganda campaign in theory becomes significantly more effective. As such, there is a vested interest in preventing propaganda attacks on these platforms.

When discussing Propaganda detection in the context of OSNs, the problem may be reduced down to the detection of malicious agents. These agents aren't exclusively defined as spreading propaganda - there is a significant overlap between the detection of propaganda bots, and other unwanted behaviour. This includes spam, false identities, misinformation/fake news, et cetera. For the detection of these agents, a wide range of information can be considered, including account details, connected users, the content of posts/comments released, or detection of similarly behaving accounts (which may indicate a presence of a 'botnet').

4.1.1 Malicious Actors

Labelling individuals as malicious actors, especially in the context of politics can be problematic. An OSN willing to censor people for controversial political opinions risks becoming an echo chamber. Furthermore, unorganised individuals voicing opinions hardly constitutes a propaganda campaign. Therefore, most efforts towards detecting and censoring propaganda in a social media setting fall under the realm of *Bot Detection*. Whilst 'bots' can generally be defined as 'automated accounts on social media', bots aren't inherently negative, especially when it is explicitly known that the account is a bot. Thus, distinctions between various types of bots should be made. Unfortunately, the current literature can't seem to agree on concrete definitions [34], with definitions overlapping or contradicting one another. This would not be a particularly pressing matter if not for the previously mentioned usage of social media bots during major political events. As such, here is (a version of) a list of some commonly used bots [34].

- **Spambots** are malicious actors with the intent of spreading information as much as possible. This can range from somewhat benign behaviour of posting advertisements, to the active spread of misinformation, or the overwhelming of online platforms. One nefarious tactic is the overwhelming of political discussion boards via spam, drowning out whichever message the bots are working against. This type of bot isn't exclusive to OSNs, and can be found on almost any platform with a commenting feature.
- **Social Bots** is the broad category of bots which aim to appear human-like. These can be used for a wide range of purposes, like spambots, aren't inherently malicious with them being potentially used for activism or journalism. **Political bots** are a subcategory of social bots, which are defined by their purpose of influencing political opinion [97]. This can be done, among other things, through smear campaigns or similarly to spambots, overwhelming discussion threads.
- **Sock Puppets**, or trolls, are fake identities used to interact with normal users on OSNs. These are distinct from social bots, as they are not necessarily automated, and may involve partial, or full human control. Whilst technically not being bots, they may behave in similar mannerisms and can still be used for the same purposes (propaganda). Due to the fact that they are not (fully) automated, they often require significantly more manpower to operate. One example is the 'Internet Research Agency' [31], a Russian organisation with the purpose of spreading propaganda during the 2016 U.S election.
- **Cyborgs** refers to any combination of automated bot and human curation, It is unclear in literature how much automation classifies a user as a cyborg. Do scheduled posts make an account a cyborg? Are automatic responses to private messages enough? Regardless, cyborgs make bot detection substantially more difficult by displaying human-like behaviour, resulting in a false-negative. The issue is further exacerbated by bot-like behaviour from non-bot accounts. For instance, human "click-workers" were hired by the Vietnamese government to "spread political messages and to like, upvote, and share content algorithms" [51].

Hopefully it is evident that the existing categories of bots are vague, overlap, and can contradict each other - for example, sock puppets are often discussed in the context of bots while technically not being automated. Nonetheless these categories paint a picture of one of the main challenges of bot detection - defining what a bot actually is. Additionally, while not all of these categories directly spread propaganda, they may all play a role in a propaganda campaign. Spambots may be used to shut down critical discussion, Social bots may be used to spread misinformation en masse. Sock puppets may distribute more sophisticated propaganda, while even minimal levels of human curation would make the other bots more difficult to classify.

4.1.2 Initial Designs

Initial approaches to propaganda/bot detection relied on individual account analysis. This was done via the use of machine learning methods such as *Support Vector Machines*, *random forests*, or *boosting/bagging*, with manually assigned features such as post rate, number of followers or number of people followed [57]. An example detection method for spambots was written by Sarah Body [101]. This paper aimed to understand the differences between actual accounts and spam accounts. Despite initial hypotheses being that certain properties would be quite distinct, such as post rate or follower/following ratio, in practice this was not the case, with only the number of followers/following being noticeably distinct in normal accounts as opposed to spam. In general, early detection methods were relatively simple relying on out-of-the-box machine learning algorithms and a small handful of hand-crafted features. These detectors had relative success, but as OSNs grew so did the sophistication of the bots. As such detection methods grew alongside this. Another example is Botometer [100], a supervised machine learning architecture for Bot detection on Twitter. Having been trained on a wide range of datasets, the system extracts over 1000 features for each inputted account to calculate a total 'legitimacy' score, as well as sub-scores for user meta-data, friends, content, sentiment, network, and timing. Including these examples, broadly, the features used for bot identification in early individual account detection can be placed into 1 of 4 categories (as is outlined by Hayawi et al. [36]):

1. **User metadata** - information that comes from the users profile such as their name, follower count, usage etc.
2. **Temporal/behavioural** - features such as post types, post timestamps.
3. **Textual/content based** - the content of the users posts, number of hashtags, types of hashtags, etc.
4. **Interaction based** - number of comments, likes, and reposts.

While these models initially had some success, such approaches have been shown to be suboptimal [18]. Firstly, humans struggle to identify more sophisticated bots on social media platforms. Studies [19] have suggested that as bots become more sophisticated, even tech-savvy individuals struggle to identify them. It therefore became increasingly more difficult to create high-quality training data with manual annotation and feature selection. Furthermore, as detection methods improved, bots became better at avoiding them, with bots quickly outgrowing individual account detection. This concept is referred to as *bot evolution*. This problem is even more prominent at the time of writing this due to the rapid improvements seen in LLMs - for instance, another study suggests that LLM generated misinformation is more difficult to identify as misinformation than human written misinformation [15]. Meanwhile, detecting LLM generated text is still proving to be a challenge [99]. Furthermore, individual account detection tends to fail if any human behaviour is displayed, allowing for partially automated bots such as *cyborgs* to run unchecked. In short, the focus on features of individual accounts resulted in bots adapting their behaviour to specifically avoid account based detection.

4.1.3 Group Analysis

More success was seen by analysing and detecting coordinated suspicious behaviour rather than individual accounts. Jiang et al. [44] proposed a detection method for twitter which looks for *synchronised* behaviour (nodes which all behave very similarly to one another) and *rare* behaviour (behaviour that is not displayed by other accounts). Another model by Mazza et al. [60] used an LSTM to convert retweet time data into feature vectors, which could then be clustered based on normal vs. malicious behaviour. Although such approaches saw more success than the individual account detection, this success was limited [18]. In practice, detections are developed in response to an evolution in botnets - this means that these bots can't be stopped outright, only suppressed after they have already done considerable damage. In other words, in the arms race between detection and bots, detection methods are always outgunned. These methodologies are therefore flawed as they fail to prevent these attacks before they occur. Cresci

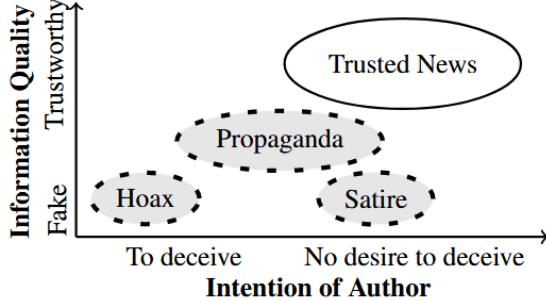


Figure 13: The classes assigned to each document by Rashkin et al. [76]

et al. [18] argue that this is due the assumption made when using machine learning - namely, that the environments they will be used for are *stationary* (data generation remains constant) which is not the case. They argue for the use of adversarial machine learning detectors. To the best of the author's knowledge, minimal development has been made in such systems since the writing of the paper.

In short, the network analysis methodology, while sound in theory, is not particularly suited for propaganda detection due to it not being able to keep up in an a non-stationary setting. Furthermore, there don't exist any quality, up-to-date datasets (to the best of the author's knowledge) as it is becoming increasingly more difficult to differentiate between bots and humans online. As such, alternate avenues of computational propaganda detection were explored.

4.2 Natural Language Processing

Whilst network analysis aims to identify bad actors in an OSN, the content of the messages is rarely examined in detail. As mentioned, content-based features of posts have been utilised previously, however in network analysis models, the features used typically quite simple. Natural language processing takes an alternate avenue by aiming to identify individual pieces of propagandist media rather than bad actors. The core idea behind this approach is that persuasion will rely on the same set of persuasive markers, regardless of how it is distributed. This comes with its own sets of challenges, and is by no means a perfect solution - when propaganda campaigns are ran, one would expect to see coordinated behaviour which can't be seen via the analysis of an individual document. Nonetheless, by identifying propagandist media rather those who spread it, the issues associated with bot evolution can be circumvented, and propaganda can be filtered after its been distributed.

4.2.1 Document Wide-Classification

Lexical Analysis

Early instances of computational propaganda detection (via NLP) began in 2017[76]. Initial approaches relied on whole document classification, with Rashkin et al. [76] using the labels of *not-propaganda*, *propaganda*, *satire* and *misinformation* with the intent to deceive and truthfulness of each article determining their class (as is shown in figure 13). The articles used in the study were collected from 8 sources, with the classification of the articles based off their respective sources, which are classified listed in US News & World Report, a reputable news outlet analysis program [88]. This is a form of *weak supervision* where the gold labels for a dataset are given via some heuristic rather than a manual annotation. The lexical features of each article were calculated by comparing the ratio of lexicon markers (sets of words) in trusted vs fake news. Some examples are: sets of *strongly subjective words*: *second person pronouns*: or use of *sexual phrases*.

Additionally, a classification task with based on n-gram tf-idf feature vectors (technique for reflecting the importance of a word in a document) was tested for the veracity of specific statements, with the same lexical features being experimented with as part of an ablation study. The classes were either: *True*, *False* or *True*, *Mostly True*, *Half True*, *Half False*, *Mostly False*, *False*. The appended features significantly improved performance in traditional ML models (Maximum Entropy classifier , Naive Bayes) while showing little improvement for a base LSTM.

Performance was found to be significantly worse (but better than a random baseline) when testing on sources the model was not trained on, suggesting that while such an approach has potential, there was a tendency to model the source of an article, rather than any propagandistic content. It is unclear whether the articles chosen were manually curated to contain propaganda, or were chosen at random. Note that this dataset was constructed under the assumption that all publishings of 'propagandist' sources are pro-

paganda, which is not in fact the case [38]. This therefore results in noisy training data where the labels more accurately reflect the sources rather than propagandistic content.

Lexicon Markers	Ratio	Source	Example Text	Max
Swear (LIWC)	7.00	<i>Borowitz Report</i>	... Ms. Rand, who has been damned to eternal torment ...	S
2nd pers (You)	6.73	<i>DC Gazette</i>	You would instinctively justify and rationalize your ...	P
Modal Adverb	2.63	<i>American News</i>	... investigation of Hillary Clinton was inevitably linked ...	S
Action Adverb	2.18	<i>Activist News</i>	... if one foolishly assumes the US State Department ...	S
1st pers singular (I)	2.06	<i>Activist Post</i>	I think it's against the law of the land to finance riots ...	S

Table 1: Some of the lexicon markers used by Rashkin et al.[76] - taken directly from the paper. Listed are their relative ratios sources, example snippets, and the outlet class in which each marker peaks (S = satire, H = hyperpartisan, P = political).

The lexicon markers with the highest correlations were usually indicators of satire, which is expected to contain over-the-top language. Interestingly, out of the top 5 lexicon markers with highest ratios compared to the baseline, only 1 is associated with propaganda, suggesting that the vocabulary of propaganda doesn't differ much to normal text.

More thorough research on the lexical features of propagandistic text was done - specifically [6] by Barron-cedeno et al. Another document-level dataset was created consisting of 10 propagandistic sources (as opposed to 2 in TSHP-17, the dataset developed by Rashkin et al.[76] and named as such by Barron-cedeno et al. [6]), and 94 non-propaganda sources. The documents collected contained additional metadata including their source, and as was done in TSHP-17, were labelled according to their sources (as is defined by Media Bias/Fact Check [61]). A handful of features were proposed; these included lexicons as had been done in past papers [76], as well as new features; vocabulary richness, readability and NELA (News Landscape) features.

Findings were mixed - when experimenting with the original 4-way classification on TSHP-17, all provided new representations failed to meaningfully generalise to out-of-domain sources. A similar pattern was also observed with *QPROP*. The general trend displayed was that modelling writing style and readability resulted in much better performance of models. At the same time, this approach also tended towards modelling sources rather than propagandistic content. This can be attributed to the fact that the data collected, whilst being from a wider range of sources, was still *weakly* labelled. Readability and stylistic content better modelled sources than propaganda.

4.2.2 Fine-grained analysis

An alternate, fine-grained approach was proposed instead. Da San Martino et al. [23] suggested labelling individual instances of rhetoric techniques rather than entire documents. An example is showcased in Figure 14. The argument presented is that by labelling instances of persuasion, more explainable systems can be crafted. As the persuasion techniques were labelled manually, they also contained significantly less noise than the article-wide labels in *TSHP-17* and *QPROP*.

Propaganda Techniques

Outlined are the techniques labelled in the proposed, fine-grained corpus. Note that these techniques aren't an exhaustive list of what is used in propaganda; for example, techniques such as *card stacking* ([47], p 1-48) require access to external sources, which isn't feasible for individual technique labelling. Furthermore, exact persuasive techniques aren't agreed upon - Wikipedia lists noticeably more than 18 techniques, with some arguably being encapsulated by others. This is discussed by the authors, and these definitions are expanded upon in subsequent papers [30].

1. *Loaded language* - Using words/phrases with strong emotional implications [95].

	<small><i>Stereotyping_name_calling_or_labeling</i></small>
1	Manchin says Democrats acted like babies at the SOTU
2	Democrat West Virginia Sen. Joe Manchin says his colleagues' refusal to stand or applaud during President Donald Trump's State of the Union speech was disrespectful and a signal that
	<small><i>Black-and-white Fallacy</i></small>
	the party is more concerned with obstruction than it is with progress.
	<small><i>Loaded language</i></small>
4	In a glaring sign of just how stupid and petty things have become in Washington these days, Manchin was invited on Fox News Tuesday morning to discuss how he was one of the only Democrats in the chamber for the State of the Union speech
	<small><i>Exaggeration</i></small>
	not looking as though Trump killed his grandma.
	<small><i>Loaded language</i></small>
6	As Manchin noted, many Democrats bolted as soon as Trump's speech ended in an apparent effort to signal
	<small><i>Exaggeration</i></small>
	they can't even stomach being in the same room as the president

Figure 14: Example of a fine-grained propaganda classification task [23]. The image showcases an example of how the training and testing data would be labelled. Note that the span of each individual label may overlap with others.

2. *Name calling or labelling* - labelling the target of the propaganda as something the consumer dislikes, or has negative associations.
3. *Repetition* - the repetition of the desired message.
4. *Exaggeration or minimization* - exaggerating a given action to make it seem more larger, serious, or in any other terms more intense than it actually is. Minimization is the opposite - making a given topic seem less significant than it may actually be.
5. *Doubt* - questioning the credibility of an entity or statement, in a critical, non-constructive manner.
6. *Appeal to fear/prejudice* - seeking to build support for an idea by instilling anxiety and panic in the population towards an alternative, possibly based on preconceived judgments.
7. *Flag-waving* - laying on a strong national feeling (or any other large group the consumer may identify themselves with) to justify or promote an action or idea.
8. *Causal oversimplification* - reducing a complicated set of circumstances to one cause for a particular issue.
9. *Slogans* - a brief and striking phrase that may include labelling and stereotyping.
10. *Appeal to authority* - Stating that a claim is true simply because a valid authority or expert on the issue supports it, without any other supporting evidence.
11. *Black-and-white fallacy, dictatorship* - Presenting two alternative options as the only possibilities, when in fact more possibilities exist.
12. *Thought-terminating cliche* - Words or phrases that discourage critical thought and meaningful discussion about a given topic.
13. *Whataboutism* - discrediting an opponent's position by charging them with hypocrisy without directly disproving their argument.
14. *Reductio ad Hitlerum* - persuading an audience to disapprove an action or idea by suggesting that the idea is popular with groups despised by the target audience. As the name suggests, a common example of the despised group is the Nazi Party.
15. *Red herring* - Introducing irrelevant material to the issue being discussed, so that everyone's attention is diverted away from the points made.
16. *Bandwagon* - attempting to persuade the target audience to join in and take the course of action because "everyone else is taking the same action".
17. *Obfuscation, intentional vagueness, or confusion* - using deliberately unclear words, so that the audience may have its own interpretation.
18. *Straw man* - when an opponent's proposition is substituted with a similar one which is then refuted in place of the original.

Articles were collected from 49 sources (13 propaganda, 36 non-propaganda). This was formally done by utilising the γ *inter-annotator agreement* [58] to determine the best fitting spans and labels. For NLP tasks where there exists no objective truth (which is the case in this instance) a ‘gold standard’ is aimed for instead. To achieve this, the standard approach is maximising some inter-annotator agreement measure. Data is given to annotators to label, and the confidence of their labels are quantified via some metric to evaluate the created dataset’s suitability. Multiple metrics exist; for this, γ is chosen instead due to its ability to handle overlap between spans, and to measure both accuracy in terms of spans, and their assigned labels/techniques.

After the annotators initially submitted their labels, a 3rd expert consoled all present annotators to create the final dataset. This approach reduced the bias of each individual annotator, which would in theory be a large issue for such a task as their political opinion would likely subconsciously prevent them from making an impartial choice. The paper also proposes various deep learning architectures for propaganda detection which inspired some of the cutting edge models [65] discussed later in this document. These include a BERT Baseline, 2 versions of BERT with an additional *sentence classification* task for improved accuracy, and as well as a custom *multi-granularity* network (detects if various granularities of the input contain propaganda and uses that information to create more accurate spans).

The dataset and subsequent tasks have multiple benefits. Firstly, the provided data is much higher quality - the labelling is done on an article-by-article level by professional annotators² with empirical measurements of the quality of data (inter-annotator agreement). This both ensures quality of labels, and due to an abstinence from weak labelling - less noise. Additionally, the proposed task is inherently more explainable than document-wide classification. It is more difficult and less helpful to explain to a user that something was flagged as propaganda due to its vocabulary, as opposed to the use of concrete persuasive techniques. There are still many limitations to this approach however. Firstly, the data is quite imbalanced; the most common class, *loaded language*, has 2,547 instances whereas the least common class, *straw man* only has 15. Secondly, while a larger range of sources was used to collect the articles, these sources don’t differ particularly in regard to political stance, with most being far-right and from news outlets based in the USA. As such a wider range of sources preferably in different languages would be more appropriate for propaganda detection. Furthermore, from a more general view of computational propaganda, journalistic text isn’t the only medium used to spread propaganda online.

4.2.3 Joint tasks

The fine-grained analysis approach was expanded upon in subsequent papers. This approach has been experimented with using a multitude of different architectures, with many models being submitted for joint tasks in SemEval [78], an international semantic analysis NLP workshop. One of the reasons for this focus on the fine-grained approach is the dataset described in the previous section - there exist very few alternatives for propaganda detection. Recurrent tasks are ran by the same authors as the fine grained analysis paper[23], typically themed around computational propaganda, or topics adjacent to it. The following section briefly discusses these papers, covering what is mentioned in them, including their focus, novel contributions to the field, and some note-worthy models.

- **SemEval 2020 Task 11 - Detection of Propaganda Techniques in News Articles;** this task closely adheres to the fine-grained task [24]. Specifically, the fine-grained analysis task is modularised into 2 sub-tasks; *Span identification* and *Span Classification*. Span identification refers to identifying the sequence of words in a sample text that form a propaganda technique. Technique classification refers to the classification of each of those spans into a possible propaganda technique. While the task is technically *multi-label* (the same span may have multiple propaganda techniques) in practice there is only 1 instance of this occurring in the provided dataset. For this specific task, the dataset is refined, grouping together techniques with a small number of occurrences. The most promising models were primarily *BERT*-based, with *Self-Learning*, *Conditional Random Fields*, and *LSTMs* also proving to be useful for improving accuracy.
- **SemEval 2021 Task 6 - Detection of Persuasion Techniques in Texts and Images;** this task focused on multimodal classification in the form of memes [29]. Similarly to its predecessor, the task was split into 3 subtasks. Typically, memes are composed of both a text and image component, which are both important in the spread of their message. The first was a multi-classification task of the text in the labelled meme into a propaganda technique. The second one was the same, with the additional requirement of a span for the technique. The third subtask asked contestants to predict technique labels based off both the image and the text of the meme. Similarly to the previous

²A Data Pro: <http://www.aiidatapro.com>

task, best performing models relied on *BERT* based architectures for all 3 subtasks. The joint paper also updates the propaganda technique list. This perspective for computational propaganda detection is incredibly important - As discussed previously, much of computational propaganda is spread through OSNs. Memes which have the potential to go viral on OSNs are highly effective mediums for propaganda [25] - their impact is theoretically more widespread in swaying opinion than articles. This would fall under the category of soft propaganda as memes are often seen as casual and therefore not trustworthy sources of information.

- **SemEval 2023 Task 3 - Detecting the Category, the Framing, and the Persuasion Techniques in Online News in a Multi-lingual Setup;** this task expanded the traditional subtasks required for propaganda detection [71]. The first subtask, *News Genre Categorization* asks to classify whether the article is *objective*, *satirical*, or an *opinion* piece. The second subtask, *Framing Detection* asks to classify the article into one or more *frames* [13]. To frame is to "select some aspects of a perceived reality and make them more salient in a communicating text, in such a way as to promote problem definition, causal interpretation, moral evaluation, and/or treatment recommendation for the item described" [32]. Some examples are *Economic*, *Capacity* and resources, *Morality* frames - in other words, whichever topic is being discussed would be tackled from these angles. The third subtask is once again Persuasion Technique Detection, although this time at a paragraph level i.e. classifying paragraphs instead of words. The provided dataset again expanded the persuasion technique taxonomy, as well as providing data for eight other languages besides English.
- **SemEval 2024 Task 4- Multilingual Detection of Persuasion Techniques in Memes;** this task builds upon the previous two by combining them[30]. The first subtask is a classification task based off the textual content of the meme. The second subtask is the same classification task, this time taking both text and image as an input. The third subtask is a binary classification of whether a meme contains a propaganda technique within it. Unlike the 2021 joint task, the task is multilingual with 3 other languages. Furthermore a hierarchical classification for persuasive techniques was proposed, with 3 broad categories being *Ethos*, *Pathos*, and *Logos*.

These subtasks provide robust, quality datasets for propaganda detection. The participating teams also develop custom models which perform significantly better than off-the-shelf approaches like LLMs or ML algorithms.

4.2.4 Adjacent fields

Propaganda detection, as defined by Da San Martino et al. [23] can be reduced to the detection of persuasive techniques. However, persuasion is not necessarily limited to propaganda, and as such, overlap between propaganda detection and other semantic analysis-based fields can be seen. Therefore, fields closely related to propaganda detection were considered in the hopes of finding some potential improvements to the best performing architectures.

Persuasiveness Detection

Computational persuasiveness (or persuasion) detection aims to identify linguistic patterns that make a text convincing to an audience. Researchers have proposed both *linguistic approaches* (using stylistic cues and lexicons) and *argumentative* approaches [7], which is in line for the general methodologies of propaganda detection. Importantly, persuasiveness detection is broader than propaganda detection: it draws on a wider set of rhetorical and psychological theories. In practice, many models for persuasion detection either explicitly incorporate argument structure (via Rhetorical Structure Theory or custom taxonomies) or implicitly learn it through large pretrained encoders. This draws from interdisciplinary theories of rhetoric: for example, integrating Aristotelian appeals (logos, pathos, ethos) or modern argumentation theory can make models more interpretable and aligned with human notions of persuasion. Emotional tone and sentiment are also key components of persuasion (pathos), so persuasiveness detection overlaps with sentiment and affective analysis. Many studies use sentiment lexicons or emotion dictionaries as proxies for persuasive style: for instance, Addawood et al. [1] used the MPQA sentiment lexicon alongside LIWC psychological categories to detect affective cues in "persuasive language". Ahmad and Laroche (2015) even applied cognitive appraisal theory to show that certainty and concreteness (emotional dimensions) correlate with persuasion effectiveness. Typically, the best performing models combine hard-coded features with deep learning (unlike the propaganda models, which typically almost exclusively rely on deep-learning).

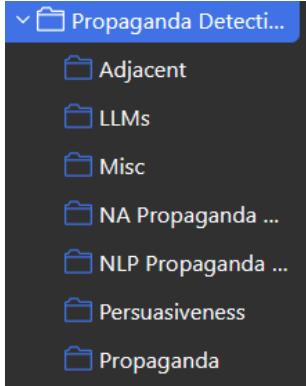


Figure 15: Folder structure of the Zotero research collection

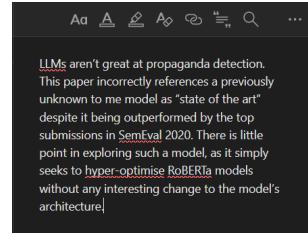


Figure 16: Example note for one of the reviewed papers

5 Methodology

For a project of this complexity, strong time management was key. As this project was heavily research based, a large amount of time was first dedicated to this. That being said, the development and evaluation processes were by no means trivial.

5.1 Research

At the beginning of the project, the scope was unclear - it took a significant amount of research to establish exactly what the aim of this project will be. This was completed over the course of 12 weeks, beginning from the summer of 2024 up until week 8 of Term 1. This research phase was also split into 2 distinct paths; research of Deep Learning/NLP, and research of Computational Propaganda Detection. It was first necessary to establish a strong foundation in the field of Machine Learning and its applications in Natural Language Processing. This was done by following through CS224N [84], a freely available Stanford University course on NLP. This course was chosen due to Stanford's well known reputation for its quality of teaching. Alongside this, Computational Propaganda Detection was researched via the use of Google Scholar [33], with the most-cited publications being prioritised. Much of this time involved learning topics ad-hoc. as they arose in sources. During research the most cited approaches were prioritised.

5.1.1 Research Management Tools

The large amount of papers being processed warranted the use of reference management. Zotero [102] is a free, open-source management software designed for this purpose. While there exist other alternatives, Zotero was chosen for its ease of use. A collection was made, separating the various papers covered by topic as is shown in figure 15. This software also allows its users to annotate the stored research, adding comments or tags. This helped summarise each paper, or leave comments for long after the paper was read.

5.2 Development

The chosen development methodology was a plan driven one - specifically, an **incremental** model. Alternative plan driven methodologies require a thorough initial system design which wasn't realistic given the lack of background in the field, and difficulty of the theory behind the models being used. It was considered more practical to design and develop these models separately in increments. An Agile methodology is also not entirely appropriate - there are no customers to provide feedback. Additionally, the robustness of the suite is a higher priority than delivery time. There is the possibility of a significant design change during development, for instance as a result of new literature being published which undermines this project, or is too significant to omit. Incremental development accommodates for changes in specification well.

For version control, Github [12] was chosen. This allowed for the suite to be developed on multiple devices, which was later key for testing specific models, and allowed rollbacks in case bugs weren't fixable. Code was developed on one branch as it wasn't necessary to isolate features - the development was linear in nature, and most of the additions were additive rather than changing existing code.

5.3 Ethical and Legal considerations

Classification of text as propaganda can be inherently problematic. Firstly, inaccurate performance may do more harm than good, labelling constructive discussion as propaganda whilst leaving more harmful content unlabelled. Additionally, the performance of the algorithm is highly dependent on the provided dataset, in which an inherent bias will exist. It is therefore very possible to train the model on unbalanced data, only labelling specific, disliked arguments as propaganda. Labelling arguments that don't fit the desired narrative as propaganda or misinformation is a not-uncommon phenomenon in political warfare. If this algorithm were to be used for such a purpose its net benefit would be greatly diminished.

The developed architectures are based on deep learning. Whilst this paradigm offers cutting-edge performance for a plethora of tasks, including this one, this comes at the cost of explainability. However, when sensitive topics such as propaganda are being discussed, explainability is key, especially for preventing bias or tampering. The individual labelling of techniques involves making less decisive claims about the text, and could be utilised for making more explainable models for classifications of entire documents in future works.

In essence, quality, unbiased performance of the architecture is crucial for its ethical use. Unfortunately, such a goal is nearly impossible to perfectly achieve - the chosen training data and its labels will always contain an implicit bias, which may only be minimised rather than eliminated. Therefore, presenting these models as unbiased supreme judges of propagandistic content would be incredibly misleading. They should therefore be treated more so like an assistive tool for other, more robust detection methods for automated systems, or as informative tools encouraging critical thinking rather than something to be blindly trusted. Nonetheless, the data chosen has been labelled in a professional, quality manner, reasonably minimising bias of any individual labeller.

The datasets being used are licenced under the Creative Commons Licence [26], allowing free use as long as the original authors are appropriately credited.

The testing and evaluation of this code was, as mentioned previously, based on SemEval 2020, for which there were many submissions. To speed up the development process some input processing helper functions were taken from a participating team - NewsSweeper. The use of their code was explicitly mentioned with, and approved by the supervisor. The repository may be found here [81]. Their contributions are clearly outlined in the provided code - however, note that none of the designs for their models were used, and although their helper functions did speed up development, a significant number of changes and customizations were made for training and testing. The submitted code outlines what was and was not designed by the author.

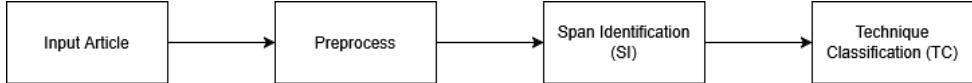


Figure 17: A hypothetical abstracted pipeline for computational propaganda detection in text. Text would be inputted and formatted via tokenization, the span of propaganda would be found, and the techniques used in each span would be identified.

6 Design and Evaluation

Based off the literature explained in the previous section an NLP approach was taken to computational propaganda detection. While datasets for a network analysis approach existed, the ones found were significantly older than those for NLP, and tended more towards bot detection than propaganda detection. Furthermore, the issue of bot evolution was discovered early, and as such research was more focused in directions which detected propaganda at the content level. The chosen dataset was from SemEval 2020, as were the main models to be developed. Newer joint papers [29, 71, 30], and as a result newer datasets tended to diverge from propaganda detection in text prioritising adjacent tasks or exploring a multilingual/multimodal setup. Although those tasks are an arguably more effective method for fine-grained propaganda detection, they were deemed to be outside the scope of the project.

The chosen models to attempt to replicate and evaluate were the parts of models from Hitachi [65], and ApplicaAI [48]. These were the best performing models for SemEval 2020 [24] for both required subtasks (*Span Identification*, *Technique Classification*). Additionally, the performance of generative AI models, specifically Deepseek R1 was evaluated. As discussed previously, Deepseek R1 is a state-of-the-art Large Language model and as such is a fantastic choice for testing generative AI for this task. However, Deepseek R1 is not possible to host locally due to its size- as such, a distilled version of QWEN 2.5, another popular Large Language Model was used instead.

The dataset contains 371 articles of varied length. For each article, 2 label files exist - one for the span identification, outlining where each propaganda snippet in the text is, and one for technique classification outlining both span and technique for each propaganda instance. Note that these are not a perfect overlap, with some spans for the technique classification files not existing in the span identification label files. It is unclear why this is the case, and such instances, when overlap between the 2 tasks was necessary, was discarded. During training, 270 articles were used for training, 30 were used for validation³, and 71 were used for testing purposes. Note that while additional validation and test labels exist, these weren't publicly available and as such the training data had to be used instead.

6.1 Languages and Tools

The entire system is written in Python [94]. Python is one of the most commonly used languages in scientific computation due to its accessible libraries and simple syntax. This however comes at the cost of performance and maintainability. While other languages offer machine learning development, such as C++, Python's accessibility and wide usage in scientific literature makes it the more appropriate choice. For developing the deep learning models, Pytorch [73] was used. Pytorch is a popular machine learning library, which allows for the creation of customisable machine learning and deep learning architectures. Most tools pertaining to encoders and LLMs sourced from the transformers library by huggingface [87]. For enriching individual word representation, SpaCy [82] was used. To test Deepseek R1/QWEN, the model was hosted locally. For this purpose Ollama [69] was used. Ollama is an LLM management system, allowing the user to host various models locally. This was key in bypassing token limitations and censorship. DCS Batch machines were primarily used for training and evaluating the custom architectures, whereas the LLM was tested on a local machine.

6.2 Preprocessing

The standard database is separated into 3 (main) sections - the articles, the technique classification labels, and span identification labels. The spans are saved in individual files with the same identification as the article, with each line containing the id of the article, the beginning character of propaganda, and end character of propaganda. When the words of the article are being parsed and tokenized, their location in the article is tracked to later output back into a prediction. Once the article has been read, its sentences are separated, which are then tokenized. Words are tokenized via *Byte Pair Encoding*, the format for *RoBERTa*. During development, various strategies were considered for tokenization - some involved adding contextual information during forward propagation of the model. While this did result in

³Initially, cross-validation was used. However, due to the long training times of these models, this was quickly discarded.

a technically functional model, it was vastly impractical to train - the training of neural networks is already costly but is mostly sped up via parallelisation in the form of tensor calculations. The token enrichment was done sequentially and thus could not be optimized, slowing down training by magnitudes, making debugging particularly difficult. Instead, additional information about tokens was extracted during the loading of the dataset. Data was stored in *tensors* ([42] p 87–104) which allowed for computationally efficient, parallelisable operations.

6.3 Baselines

Before developing the custom top-performing models it is first necessary to develop baselines for the sake of comparison. Firstly, the metrics for scoring should be discussed. The score for *span identification* is a modified F1 score, which is calculated as follows:

$$F_1(S, T) = 2 \cdot \frac{Precision(S, T) \cdot Recall(S, T)}{Precision(S, T) + Recall(S, T)} \quad (22)$$

where S is defined to be the predicted spans of all articles for in a dataset D . Similarly, T can be defined as set of the gold labels for the articles in D . In other words, $S = \{S_d\}_d$ and $T = \{T_d\}_d$, where $T_d = \{t_1, \dots, t_m\}$ and $S_d = \{s_1, \dots, s_m\}$, gold spans and predicted spans for article d in D .

Precision can be anecdotally thought of as the quality of the prediction of a model. In other words, if a model makes a prediction, how likely is it to be correct. More formally, precision is defined as

$$P(S, T) = \frac{1}{|S|} \cdot \sum_{d \in D} \sum_{s \in S_d, t \in T_d} \frac{|s \cap t|}{|t|}. \quad (23)$$

Recall is the metric for how often a model identifies true positives from all actual positives. Formally, it is:

$$R(S, T) = \frac{1}{|T|} \cdot \sum_{d \in D} \sum_{s \in S_d, t \in T_d} \frac{|s \cap t|}{|s|} \quad (24)$$

Note than in both equations (2) and (3), partial overlap is less heavily penalised (due to the use of the "and" operator). As such, partially correct spans get a non-zero F1 score. Both precision and recall are important and as such the F1 score can be accurately used to measure the quality of a model.

The score for *Technique Classification* is a standard micro-average F1 score, which can be calculated by finding the number of true positives, false positives, true negatives and false negatives. While technically the task is a multi-label one i.e. multiple labels may be assigned to the same span, only one label was assigned to each span for simplicity. The dataset does not contain a significant number of overlapping spans.

6.3.1 Large Language Models

Large Language Models are fantastic few shot learners. Their primary benefit is the ability to excel at a wide range of tasks, potentially saving valuable time and resources training a model. This does come at the cost of runtime efficiency - LLMs are expensive to run regularly, most typically being used by an external API. Given the current task this is not an option - censorship would result in suboptimal, or even worse, biased outputs. Furthermore LLMs are prone to hallucinations (making up data), inconsistent outputs, and inherent bias in their training data - the time that is saved in development may potentially be offset by the post-processing/fine-tuning that is necessary for an LLM to function well. Due to their ease of use and power, a distilled version of Deepseek R1 was used for both span identification and technique classification.

Span Identification

Initially the following prompt was given to the distilled Deepseek R1 model: *You are a propaganda-span tagger. For the article output each span on its own line exactly as: <start_char_pos><TAB><end_char_pos>. Do not output anything else. an example output is:...*

Unfortunately the output for this was nigh-unusable - the LLM often ignored the instructions, outputting full sentences which could not be parsed effectively by the scoring system with manual filtering. By utilising the set format function in Ollama [69], the output could be forced to be in an acceptable format. This still required manual curation - often, the outputted data was still invalid e.g. beginning character of the span after the end character, or characters longer than the provided text. As such the data had to be manually filtered. The tested system performed better than the baseline provided in SemEval 2020 - furthermore, providing examples did improve the performance of the model, although to a limited

Albert is going with Max B Planc to Los Angeles



Figure 18: an example of IOB tagging [59]

extent. The model had a habit of taking these examples too literally and returning them as outputs for other articles. As these models do not (necessarily) require training, no validation score was calculated - instead, a direct comparison of their performance on the test set can be done.

Model	FLC F1 (Test)	Precision (Test)	Recall (Test)
LLM (zero-shot)	0.055966	0.147084	0.034558
LLM (example format)	0.077676	0.160319	0.051255
LLM (one shot)	0.068	0.32	0.038

The zero-shot LLM was provided the listed prompt, the example format LLM was given a random, non-actual list of spans, and the one-shot model was given an example list of spans and article from the training set. Note that the precision is consistently higher than recall, suggesting that the model has a habit of undershooting the number of predictions. They perform better than the randomly generating baseline provided in SemEval 2020 [24] (F1 score of 0.31). But still far too poorly for use in any practical setting⁴. Nonetheless some credit should be assigned to the ease of the use of this approach - developing the BERT based architectures took months of work whereas the code for this was written in a much shorter timespan.

Technique Classification

Technique classification involved feeding the LLM existing spans of propaganda, and asking it to label as one of the 14 classes.

Model	Micro F1 (Test)
LLM (zero-shot)	0.16

The list of the techniques is provided without their definitions - as such the model is free to interpret their exact definitions (which is not unreasonable given the large amount of information known by the model). Note that a higher F1 score can be achieved by assigning each span *Loaded Language* (≈ 0.25), which is the most common class in the dataset.

6.4 Custom architectures

6.4.1 Span Identification

The task of Span Identification can be handled by assigning labels to each word - *Beginning*, *Inside*, and *Outside* of the span of the propaganda technique, also known as *BIO/IOB* Tagging. Other chunking⁵ labelling standards exist, however *BIO* tagging is considered the de facto approach for this due to its simplicity. Therefore, the models were trained to label each word in the text as one of the BIO tags. The first token in a propaganda sequence is labelled with a *B*, the continued words are labelled as *I*, and any words not in the sequence are labelled as *O*. This is equivalent to a 3-way classification task. Note that words may be tokenized into multiple sub-tokens - to effectively train the model only the first sub-token of each word were considered during loss calculation.

Embedding Representation

First, each article in the database must be split into sentences which shall be passed into the model. These sentences must be split into tokens. Once each word has been split into one or more tokens, they are given a numeric value by a tokenizer which acts as their id. These ids are passed into an embedding model - in this instance, the embedding model chosen is *BERT/RoBERTa*. The model also utilises a *layer-wise attention* mechanism over the hidden states *BERT* - rather than just taking the final layer of the model as the output, each of its transformer layers is weighed and summed for the final representation, with the weights being learnable parameters (its use was inspired by team Hitachi [65]). Finally, each

⁴Some tests were done on the use of LLMs for span identification with significantly higher F1 scores - this data was later discovered to be inaccurate due to an incorrectly performing scoring system.

⁵Chunking is defined as grouping non-overlapping sequences of text.

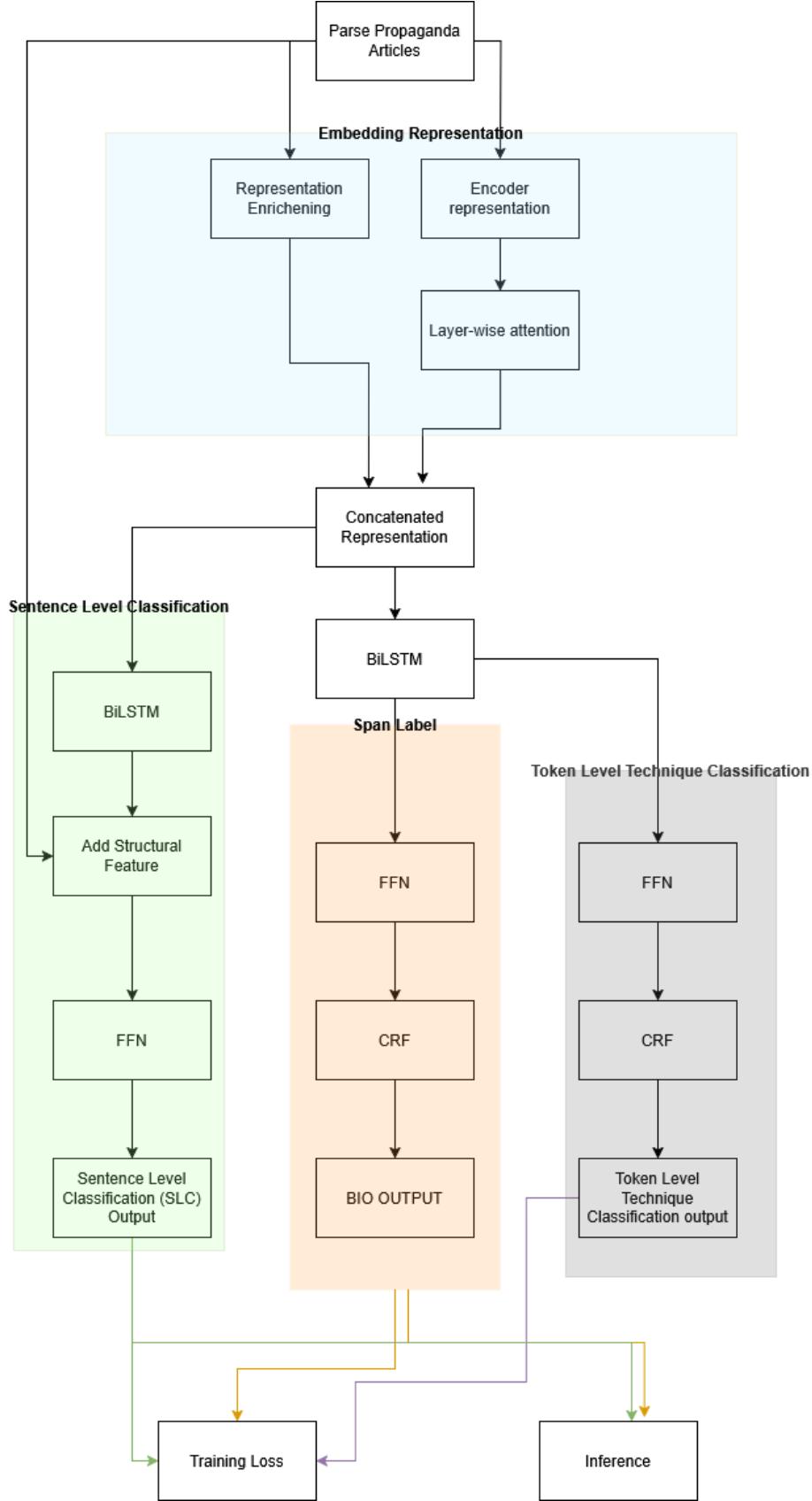


Figure 19: A high level overview of Span Identification model

word is given a *Part of Speech* and *Named Entity* tag. Part of Speech tagging [56] is a task involving assigning a label each word in an input a part of speech e.g. noun, verb, auxiliary etc. Named Entity tagging [53] involves identifying entities e.g. people, places, organizations etc. in a sentence. These are generated as *one-hot vectors* - vectors which contain 0s for all but 1 dimension. Once identified, these are

concatenated with the token embedding generated by *RoBERTa*, thereby providing more context and in certain cases improving performance.

Multi-Task learning

Multi-task learning is a paradigm in which a model is trained for multiple tasks and therefore uses multiple losses simultaneously. Paradoxically, by learning for multiple tasks as opposed to one, accuracy can improve significantly. In other words, by selecting specific tasks with overlapping needs, the model potentially generalises better. For this model, 3 tasks are trained for; *Span Identification*, the main task, *Token Level Technique Classification* - assigning each token a propaganda technique, and *Sentence Level Classification* - classifying if a sentence contains propaganda or not. The overall loss function is shown below. The loss function for Sentence level classification is Cross Entropy [20] (appropriate for binary classification). The other sub-models utilised CRF (negative log likelihood) loss [67].

$$\mathcal{L}^{(si)} = \mathcal{L}^{(si_sent)} + \begin{cases} 0 & \text{if input contains no propaganda,} \\ \mathcal{L}^{(si_bio)} + \lambda \mathcal{L}^{(si_tech)} & \text{else.} \end{cases} \quad (25)$$

Loss function as described in Hitachi, which was recreated using Pytorch. λ controls the importance of the technique classification task, and is manually tuned based off validation loss.

Sentence Level Classification

To perform sentence level classification, an LSTM is used, the outputs of which are fed through a Feed Forward Network for a binary classification i.e. *propaganda* or *not-propaganda*. This is done via processing of a special [CLS] token which is attached to beginning of any *BERT* tokenized sentence - it is used to represent the entire sentence as a whole. Two different components of the model are then responsible for similar but distinct tasks - detecting if a sentence has propaganda in it, and labelling the specific propaganda instance in a sentence. By separating these tasks, this in theory allows for each component to learn better by focusing on one specific task - in part because the span identification model isn't as impacted by the class imbalance ($\approx 75\%$ of sentences don't have any propaganda).

Main Task

The RoBERTa generated embeddings are fed through an LSTM, Feed Forward Network and Conditional Random Field to predict the *IOB* labels of each token in a sentence. During inference (i.e. when the model is asked to do a task), the spans predicted by this part of the model are only passed through if the Sentence level classification labels the sentence as containing propaganda.

Token Level Technique Classification

Token level technique classification involves assigning each token in a sentence a propaganda technique level. The same BERT-LSTM pipeline as the one for the main task is used, with a slightly different FFN head as well as a different CRF. This task teaches the model to consider not just whether each token in a sentence is propaganda, but what specific instance of propaganda it is.

Hyperparameters

Listed are the hyperparameters used for the model. There is no clear distinction between 'good' or 'bad' hyperparameters, as in practice these are tweaked to optimise the validation loss of a model on a specific dataset. The chosen hyperparameters are mostly the ones outlined in the original paper [65], with minor changes between special tag embedding dimensions. dim refers to the dimension size of each feed-forward network (the number of neurons at each layer), and λ being the scaling factor for the technique classification task. Gradient clipping is also applied - after each backpropagation step, the gradient is capped (in this case, at 1) to avoid overly large gradients causing issues during training. Layer dropout is applied to RoBERTa to prevent overfitting - it is a technique where at every iteration, a random set percent of the weights are masked as 0 so that the model does not become overly reliant on a small subset of the parameters. First epochs are frozen for RoBERTa to stabilise training results.

Results

Model	FLC F1 (Validation)	FLC F1 (Test)
RoBERTa (Large)	0.35	0.46
RoBERTa-LSTM	0.33	0.31
Multi-task Learning system	0.38	0.41

Hyperparameter	SI
Contextual embedding dim	
BiLSTM dim, layers	600, 2
PLM layer dropout	0.1
FFN ^(si_bio) dim	200
FFN ^(si_tech) dim	200
FFN ^(si_sent) dim	500
FFN activation	ReLU
λ	0.5
Adam β_1, β_2	0.9, 0.999
gradient clipping	1
epochs	10
PLM frozen first epochs	2
batch size	16

Table 2: Primary hyperparameter values for the custom span identification model

These scores are a significant improvement over the performance of standard LLMs as a result of fine-tuning. Both models were trained until no more improvement in validation loss could be seen, then ran on the test set. The performance of these models is noticeably worse than the submission to SemEval 2020 it was inspired by - this is in part due to significantly less training data (100 less articles), and lack of optimisation. Nonetheless the expected pattern is observed - RoBERTa provides a strong baseline. Despite requiring significantly more setup no significant improvement was observed with the custom models with an out-of-the-box *RoBERTa* model performing as well while being significantly easier to develop. Unfortunately, these scores highlight the difficulty in accurately identifying propaganda in text - the best performing models [65, 48] only manage to get scores of ≈ 0.50 - Far too inaccurate for reliant use. As such, proposed improvements focused on increasing this score.

6.5 Technique Classification

Slightly more simple models were developed for technique classification, relying almost entirely on RoBERTa. The first developed model was a standard *RobertaForSequenceClassification*, which pools the generated embeddings to return a single label for a sequence of tokens. This was applied directly over the propaganda spans. Another model was also developed to address the large class imbalance seen in the training dataset: The model developed for classification is an ensemble of fine-tuned RoBERTa st composed of 2 models. The voting system chosen is *majority voting*.

Model	Micro F1 (Validation)	Micro F1 (Test)
RoBERTaForSequenceClassification	0.67	0.66
RoBERTaForSequenceClassification (reweighted)	0.44	0.43
Ensemble	-	0.66

Class imbalance

As mentioned before, the provided data is heavily imbalanced, as can be seen in Figure 20. Training data like this can result in models prioritising the most frequent classes during training while ignoring the less frequent ones. To combat this, the loss function can be adjusted to be more impactful if a rare technique is misclassified as opposed to a common technique - this way in theory the model learns all techniques involved.

Ensembles

Often models perform better in groups. As such, the 2 developed models were placed into an ensemble, which was done via *soft voting* - the probabilities of each label were averaged between the two models and the most likely label was taken as the output. The scores can be seen in the table above - unfortunately, the re-weighted RoBERTa model does not seem to contribute to the ensemble in any meaningful way. Experimentation with various versions of *BERT* in an ensemble setting would likely yield better results.

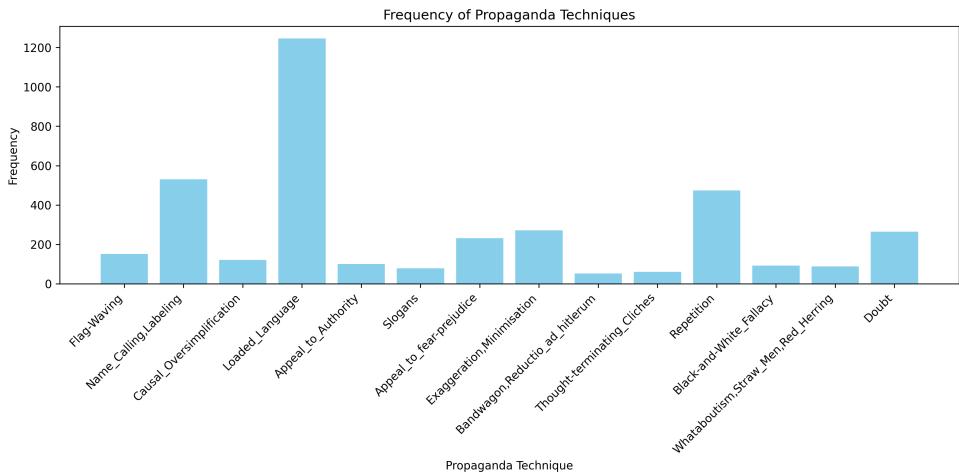


Figure 20: The frequencies of each of technique label in training data.

Hyperparameter	TC
AdamW Learningrate	1.9e-5
gradient clipping	1
epochs	10
batch size	16

Table 3: Hyperparameter values for the technique classification models

6.6 Considered approaches

1. *Self-Learning;* Self learning is a technique in which a model, in part, trains itself. In this instance, the approach involves training off the gold data, then labelling a large corpus of text [70] thereby creating a larger, albeit lower quality dataset. This was used with great success by ApplicaAI [48]. Note that their approach involved the use of Self-Learning on a single *RoBERTa* model, whereas in this instance, it would be applied to the custom model inspired by team Hitachi [65]. Despite promising initial tests, it was found to be too slow to retrain the span identification model. This approach also works significantly worse in regards to technique classification due to the inaccuracies in the self-labelled spans polluting the training data too heavily. Additionally, using this large corpus in conjunction with the hardware needed to train these models in a reasonable time was found to be a challenge. As such, this approach was abandoned - given more time and resources, this would have been the prioritised improvement for the Span Identification model.
2. *Semantic Features;* Various semantic features were considered as is outlined in the progress report. These, among others, included sentence-wide persuasion measuring features [90] (Speech Style, Lexical Complexity, Concreteness, Subjectivity) as well as NELA features. NELA [66] features are groups of text-based features for news veracity detection, a task closely related to propaganda detection. These could have been used for both span identification and technique classification - specifically, adding extra information to the [CLS] token of each sentence or propaganda span.
3. *Data Augmentation;* - instead of *self-training*, additional training data can be created via augmentation of existing data. One example is the translation of text into another language, then back to the original - this could in theory be used to create more propaganda spans/articles for the model to train on. LLMs can also be used to generate new training data via prompting. A similar strategy was used by team 914isthebest [52] for SemEval 2024, where the textual content of the provided memes was augmented using large language models to generate more training data. These approaches do come at the risk of creating poor quality data, however an exploration of this approach would have been informative.

Table 4: Micro average F1 Scores for individual propaganda techniques on the test set.

Technique	F1 Score
Appeal to Authority	0.4706
Appeal to fear-prejudice	0.3662
Bandwagon, Reductio ad Hitlerum	0.3529
Black-and-White Fallacy	0.0000
Causal Oversimplification	0.4793
Doubt	0.6486
Exaggeration, Minimisation	0.5829
Flag-Waving	0.7184
Loaded Language	0.7749
Name Calling, Labeling	0.7689
Repetition	0.0882
Slogans	0.4186
Thought-terminating Cliches	0.1860
Whataboutism, Straw Men, Red Herring	0.2143

7 Conclusion

7.1 Project Management

The outlined objectives were (mostly) met. An investigation into computational propaganda detection was conducted, and a description of the current field was written as part of this document. Some top performing models from a computational propaganda task were developed, although due to difference in implementation details the models did have some differences which overall decreased the performance. These models were compared to an LLM baseline, their performance on the validation and test data was discussed. In good conscience however, it would be unfair to claim that a novel architecture had been proposed - the explored methods were primarily inspired by existing models. Nonetheless, potential improvements to these models were listed, and given more time would have been developed. Initially a goal of 5 models (both Span Identification and Technique Classification from both teams Hitachi [65] and ApplicaAI [48], and Technique Classification by team [52]). It became obvious that this would not be feasible when development had began - the learning curve for the required libraries was much greater than expected, and many unexpected errors were ran into which delayed development times. As such, the system was scaled back to prioritise the best performing models for each task. In practice 3 models based off the examples from SemEval 2020 were (mostly) developed, albeit with the Self-Learning features outlined by [48] being omitted due to the amount of time it would have taken to train those models. The objectives were intentionally left open-ended to allow for further explorations if time had allowed, or conversely a limited set of improvements - the latter approach was taken.

7.2 Literature review and future work

Based off the survey of computational propaganda detection, the following conclusions may be derived.

Conclusion 1

Firstly, there exists a disconnect between the literature and categorization of propaganda (including computational propaganda) done in social sciences, and the definitions used in Computer Science. This may potentially be bottlenecking the performance of computational propaganda detection systems which at the current moment, lack a set definition of what propaganda is (besides the persuasive techniques being classified in various SemEval tasks). As such, research into alternate tasks for computational propaganda should be considered and explored - these would likely benefit with consultation of the various categories of bots or propaganda outlined in the current literature (and briefly described in this document). By providing more concrete definitions of propaganda more robust and better-performing systems may be developed. The correct approach seems to indicate an attitude in detection literature of propaganda being one entity, when in fact social sciences suggest the opposite. Nonetheless, the current detection methods are limited in terms of accuracy and as such may benefit from narrowing the scope of the problem. Any such further explorations would have to pick their selected features carefully, as the problem of bot evolution [18] would still be an issue. By retaining a focus on persuasiveness detection, while also narrowing down the techniques/ features indicative of various categories of propaganda, propaganda may be detected more robustly regardless of bot evolution.

Conclusion 2

Secondly, the current tools provided for computational propaganda detection aren't enough, and future works should be actively conscious of the implicit and explicit biases displayed by tools such as Large Language Models. As discussed, leading AI startups (OpenAI, Anthropic, Deepseek) are corporate entities with their own values, which may clash with the classification of specific text as propaganda. This has been briefly showcased by the censorship seen in some of the models - in future works it would be beneficial to explore this topic further beyond the surface level mention it received in this document. As such further research into computational propaganda detection must carefully consider the use of systems with an in-built bias, even if their performance is shown to be significantly better than smaller models. Note that a bias is impossible to avoid - regardless of the developer, any model will contain an implicit bias as a result of the data it has been trained on, which may contain unsavoury or even immoral content. Nonetheless, more computationally efficient systems which can be ran locally by an average user, as opposed to an architecture which relies on external API calls may be the more sensible route, even if accuracy suffers.

Self-reflection

A more concrete literary review approach would have likely both sped up the process, and improved the scope and depth of the report. Given that there was no prior experience to in-depth scientific literary reviews, nor were the researched fields something the author had any prior experience with, the research

collected was sufficient. Given the timeline of the project, less time initially spent on research would have perhaps given more leeway in the latter half of the project. However there was no way of guaranteeing that the chosen direction was in fact the best option - as such various approaches had to be considered. In the end, a focus on more modern systems for Computational Propaganda Detection (e.g. evaluation of SemEval 2024 rather than 2020) would have resulted in a more relevant project.

7.3 Development

Two specialised sub-models for computational propaganda detection in text were developed and evaluated. The utility of Large Language Models (Deepseek R1) was also explored. Then, some potential architectural improvements were proposed. Development was a challenging task - the main challenge was the disconnect between the theory of deep learning/NLP and implementations of models. This led to poor design choices which made testing/evaluation of the systems significantly more difficult than they could have been. Nonetheless the models were developed and evaluated. Future work would primarily involve adding more features to evaluate/experiment with, as well as exploring various architectural changes - for instance, adjusting the ensemble method used for technique classification.

Hyper-parameter optimization

Given the time-frame of the project, and lack of knowledge going in, there was little time for an exploration of hyperparameter adjustments. Most were taken directly from the papers the models were based off which resulted in adequate performance - however, due to differences in implementation it is possible that these weren't the optimal hyper-parameters for this task.

LLM improvement

The LLM baseline provided shows that fine-tuned and trained models outperform standard, zero and one-shot LLMs. However, LLMs can also be fine-tuned. It is significantly more computationally expensive to do so as the number of trainable parameters is magnitudes larger than those of BERT/RoBERTa/LSTMs. One technique commonly used to handle this is *Parameter Efficient Fine Tuning*, or PEFT which involves training a small subset of the LLMs parameters for a specific task, improving performance in a computationally efficient manner.

Statistical Analysis

Other papers used statistical analysis methods [48] to more conclusively determine if components of a model were improving its performance. Specifically, the Spearman Rank Correlation Coefficient [83] was used to measure the significance of various alterations to the model. In future work, such an approach would provide a much more robust conclusion - the current approach is susceptible to imbalances in data.

7.4 Self-reflection

The span identification model was found to be problematic in development due to its many non-standard approaches, as well as lack of experience. The model was initially designed with the calculation of NER and PoS tags during forward propagation - at that point in time it was not clear how computationally expensive these operations were. This drastically slowed down training time and made the task of improving the model significantly more difficult than needed, with the redesign also eating into evaluation time. Various other bugs were encountered during development in relation to settings not known to be important - a large amount of time was wasted due to the fact that data-loader was not shuffled during training, which resulted in overfitting. Nonetheless most of these bugs were dealt with (although the performance of the Span Identification model implies that the data wasn't being loaded entirely correctly).

7.5 Final thoughts

This project was both incredibly challenging and incredibly rewarding - it has forced me to push myself in both understanding, development of Deep Learning systems. Despite its ups and downs, I am incredibly glad to have chosen this project for my dissertation. It has sparked a passion for Natural Language Processing, which I am hoping to pursue in any future academic and professional career.

References

- [1] Aseel Addawood et al. "Linguistic Cues to Deception: Identifying Political Trolls on Social Media". en. In: *Proceedings of the International AAAI Conference on Web and Social Media* 13 (July 2019), pp. 15–25. ISSN: 2334-0770, 2162-3449. DOI: 10.1609/icwsm.v13i01.3205. URL: <https://ojs.aaai.org/index.php/ICWSM/article/view/3205> (visited on 11/03/2024).
- [2] AIML.com. *What do you mean by Sequence data? Discuss the different types.* en-US. Aug. 2023. URL: <https://aiml.com/what-does-sequential-data-mean-which-models-are-best-suited-for-handling-sequential-data/> (visited on 04/16/2025).
- [3] Jay Alammar. *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning).* URL: <https://jalammar.github.io/illustrated-bert/> (visited on 04/23/2025).
- [4] *Antisemitism - Photograph.* en. URL: <https://encyclopedia.ushmm.org/content/en/gallery/antisemitism-photographs> (visited on 04/22/2025).
- [5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization.* en. arXiv:1607.06450 [stat]. July 2016. DOI: 10.48550/arXiv.1607.06450. URL: <http://arxiv.org/abs/1607.06450> (visited on 04/23/2025).
- [6] Alberto Barrón-Cedeño et al. "Proppy: Organizing the news based on their propagandistic content". en. In: *Information Processing & Management* 56.5 (Sept. 2019), pp. 1849–1864. ISSN: 03064573. DOI: 10.1016/j.ipm.2019.03.005. URL: <https://linkinghub.elsevier.com/retrieve/pii/S0306457318306058> (visited on 10/07/2024).
- [7] Davide Bassi, Søren Fomsgaard, and Martín Pereira-Fariña. "Decoding persuasion: a survey on ML and NLP methods for the study of online persuasion". en. In: *Frontiers in Communication* 9 (Oct. 2024), p. 1457433. ISSN: 2297-900X. DOI: 10.3389/fcomm.2024.1457433. URL: <https://www.frontiersin.org/articles/10.3389/fcomm.2024.1457433/full> (visited on 11/03/2024).
- [8] Yoshua Bengio et al. "A Neural Probabilistic Language Model". en. In: () .
- [9] James R. Bennett, Edward S. Herman, and Noam Chomsky. "Manufacturing Consent: The Political Economy of the Mass Media." en. In: *Contemporary Sociology* 18.6 (Nov. 1989), p. 937. ISSN: 00943061. DOI: 10.2307/2074220. URL: <http://www.jstor.org/stable/2074220?origin=crossref> (visited on 10/07/2024).
- [10] Alessandro Bessi and Emilio Ferrara. "Social bots distort the 2016 U.S. Presidential election online discussion". In: (2016). URL: <https://doi.org/10.5210/fm.v21i11.7090>.
- [11] Tom B. Brown et al. *Language Models are Few-Shot Learners.* en. arXiv:2005.14165 [cs]. July 2020. DOI: 10.48550/arXiv.2005.14165. URL: <http://arxiv.org/abs/2005.14165> (visited on 04/24/2025).
- [12] *Build software better, together.* en. URL: <https://github.com> (visited on 04/07/2025).
- [13] Dallas Card et al. "The Media Frames Corpus: Annotations of Frames Across Issues". en. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: Association for Computational Linguistics, 2015, pp. 438–444. DOI: 10.3115/v1/P15-2072. URL: <http://aclweb.org/anthology/P15-2072> (visited on 04/11/2025).
- [14] Alex Carey. *Taking the Risk Out of Democracy: Corporate Propaganda Versus Freedom and Liberty.* en. Google-Books-ID: _UrtQ89Kp3YC. University of Illinois Press, 1997. ISBN: 978-0-252-06616-0.
- [15] Canyu Chen and Kai Shu. *Can LLM-Generated Misinformation Be Detected?* en. arXiv:2309.13788 [cs]. Apr. 2024. URL: <http://arxiv.org/abs/2309.13788> (visited on 10/21/2024).
- [16] Chu-yuan Cheng. *Behind The Tiananmen Massacre: Social, Political, And Economic Ferment In China.* New York: Routledge, Sept. 2019. ISBN: 978-0-429-03362-9. DOI: 10.4324/9780429033629.
- [17] Aakanksha Chowdhery et al. *PaLM: Scaling Language Modeling with Pathways.* en. arXiv:2204.02311 [cs]. Oct. 2022. DOI: 10.48550/arXiv.2204.02311. URL: <http://arxiv.org/abs/2204.02311> (visited on 04/28/2025).
- [18] Stefano Cresci. "A decade of social bot detection". en. In: *Communications of the ACM* 63.10 (Sept. 2020), pp. 72–83. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/3409116. URL: <https://dl.acm.org/doi/10.1145/3409116> (visited on 10/20/2024).
- [19] Stefano Cresci et al. "The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race". en. In: *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion.* arXiv:1701.03017 [cs]. 2017, pp. 963–972. DOI: 10.1145/3041021.3055135. URL: <http://arxiv.org/abs/1701.03017> (visited on 11/25/2024).

- [20] *CrossEntropyLoss* — PyTorch 2.7 documentation. URL: <https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html> (visited on 04/30/2025).
- [21] CS331 Neural Computing. URL: <https://warwick.ac.uk/fac/sci/dcs/teaching/material/cs331/> (visited on 04/16/2025).
- [22] Scott M. Cutlip. *Effective public relations*. eng. Upper Saddle River, N.J. : Pearson Prentice Hall, 2006. ISBN: 978-0-13-008200-8 978-0-13-123014-9. URL: http://archive.org/details/effectivepublicr0000cutl_i5p1 (visited on 04/29/2025).
- [23] Giovanni Da San Martino et al. “Fine-Grained Analysis of Propaganda in News Article”. en. 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)*. Hong Kong, China: Association for Computational Linguistics, 2019, pp. 5635–5645. DOI: 10.18653/v1/D19-1565. URL: <https://www.aclweb.org/anthology/D19-1565> (visited on 10/07/2024).
- [24] Giovanni Da San Martino et al. “SemEval-2020 Task 11: Detection of Propaganda Techniques in News Articles”. en. In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, 2020, pp. 1377–1414. DOI: 10.18653/v1/2020.semeval-1.186. URL: <https://aclanthology.org/2020.semeval-1.186> (visited on 10/07/2024).
- [25] Julia R. DeCook. “Memes and symbolic violence: #proudboys and the use of memes for propaganda and the construction of collective identity”. In: *Learning, Media and Technology* 43.4 (Oct. 2018). Publisher: Routledge .eprint: <https://doi.org/10.1080/17439884.2018.1544149>, pp. 485–504. ISSN: 1743-9884. DOI: 10.1080/17439884.2018.1544149. URL: <https://doi.org/10.1080/17439884.2018.1544149> (visited on 05/02/2025).
- [26] *Deed - Attribution 4.0 International - Creative Commons*. URL: <https://creativecommons.org/licenses/by/4.0/> (visited on 04/07/2025).
- [27] DeepSeek-AI et al. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*. en. arXiv:2501.12948 [cs]. Jan. 2025. DOI: 10.48550/arXiv.2501.12948. URL: <https://arxiv.org/abs/2501.12948> (visited on 05/12/2025).
- [28] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. en. arXiv:1810.04805 [cs]. May 2019. DOI: 10.48550/arXiv.1810.04805. URL: <https://arxiv.org/abs/1810.04805> (visited on 11/24/2024).
- [29] Dimitar Dimitrov et al. “SemEval-2021 Task 6: Detection of Persuasion Techniques in Texts and Images”. en. In: *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*. Online: Association for Computational Linguistics, 2021, pp. 70–98. DOI: 10.18653/v1/2021.semeval-1.7. URL: <https://aclanthology.org/2021.semeval-1.7> (visited on 04/11/2025).
- [30] Dimitar Dimitrov et al. “SemEval-2024 Task 4: Multilingual Detection of Persuasion Techniques in Memes”. en. In: *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. Mexico City, Mexico: Association for Computational Linguistics, 2024, pp. 2009–2026. DOI: 10.18653/v1/2024.semeval-1.275. URL: <https://aclanthology.org/2024.semeval-1.275> (visited on 10/24/2024).
- [31] Renee DiResta et al. “The Tactics & Tropes of the Internet Research Agency”. en. In: () .
- [32] Robert M. Entman. “Framing: Toward Clarification of a Fractured Paradigm”. en. In: *Journal of Communication* 43.4 (Dec. 1993), pp. 51–58. ISSN: 0021-9916, 1460-2466. DOI: 10.1111/j.1460-2466.1993.tb01304.x. URL: <https://academic.oup.com/joc/article/43/4/51-58/4160153> (visited on 04/11/2025).
- [33] Google Scholar. URL: <https://scholar.google.com/> (visited on 04/07/2025).
- [34] Robert Gorwa and Douglas Guilbeault. “Unpacking the Social Media Bot: A Typology to Guide Research and Policy”. en. In: *Policy & Internet* 12.2 (June 2020), pp. 225–248. ISSN: 1944-2866, 1944-2866. DOI: 10.1002/poi3.184. URL: <https://onlinelibrary.wiley.com/doi/10.1002/poi3.184> (visited on 04/08/2025).
- [35] Daniel Graupe. *Principles of artificial neural networks*. en. 2. ed., repr. Advanced series on circuits and systems 6. New Jersey: World Scientific, 2008. ISBN: 978-981-270-624-9.
- [36] Kadhim Hayawi et al. “Social media bot detection with deep learning methods: a systematic review”. en. In: *Neural Computing and Applications* (Mar. 2023). ISSN: 0941-0643, 1433-3058. DOI: 10.1007/s00521-023-08352-z. URL: <https://link.springer.com/10.1007/s00521-023-08352-z> (visited on 04/02/2025).

- [37] Nguyen Phong Hoang et al. “How Great is the Great Firewall? Measuring China’s {DNS} Censorship”. en. In: 2021, pp. 3381–3398. ISBN: 978-1-939133-24-3. URL: <https://www.usenix.org/conference/usenixsecurity21/presentation/hoang> (visited on 04/29/2025).
- [38] Benjamin Horne, Sara Khedr, and Sibel Adali. “Sampling the News Producers: A Large News and Feature Data Set for the Study of the Complex Media Landscape”. en. In: *Proceedings of the International AAAI Conference on Web and Social Media* 12.1 (June 2018). ISSN: 2334-0770, 2162-3449. DOI: 10.1609/icwsm.v12i1.14982. URL: <https://ojs.aaai.org/index.php/ICWSM/article/view/14982> (visited on 04/04/2025).
- [39] “How to Detect Propaganda”. en. In: *Institute for Propaganda Analysis* 1 (1938).
- [40] Philip N. Howard and Bence Kollanyi. “Bots, #StrongerIn, and #Brexit: Computational Propaganda During the UK-EU Referendum”. en. In: *SSRN Electronic Journal* (2016). ISSN: 1556-5068. DOI: 10.2139/ssrn.2798311. URL: <https://www.ssrn.com/abstract=2798311> (visited on 11/21/2024).
- [41] Zhiheng Huang, Wei Xu, and Kai Yu. *Bidirectional LSTM-CRF Models for Sequence Tagging*. en. arXiv:1508.01991 [cs]. Aug. 2015. DOI: 10.48550/arXiv.1508.01991. URL: <http://arxiv.org/abs/1508.01991> (visited on 04/29/2025).
- [42] Sagar Imambi, Kolla Bhanu Prakash, and G. R. Kanagachidambaresan. “PyTorch”. en. In: *Programming with TensorFlow: Solution for Edge Computing Applications*. Ed. by Kolla Bhanu Prakash and G. R. Kanagachidambaresan. Cham: Springer International Publishing, 2021, pp. 87–104. ISBN: 978-3-030-57077-4. DOI: 10.1007/978-3-030-57077-4_10. URL: https://doi.org/10.1007/978-3-030-57077-4_10 (visited on 05/13/2025).
- [43] *Internet and social media users in the world 2025*. en. URL: <https://www.statista.com/statistics/617136/digital-population-worldwide/> (visited on 04/08/2025).
- [44] Meng Jiang et al. “Catching Synchronized Behaviors in Large Networks: A Graph Mining Approach”. en. In: *ACM Transactions on Knowledge Discovery from Data* 10.4 (July 2016), pp. 1–27. ISSN: 1556-4681, 1556-472X. DOI: 10.1145/2746403. URL: <https://dl.acm.org/doi/10.1145/2746403> (visited on 04/28/2025).
- [45] *John Rupert Firth*. en. Page Version ID: 1281528392. Mar. 2025. URL: https://en.wikipedia.org/w/index.php?title=John_Rupert_Firth&oldid=1281528392 (visited on 04/24/2025).
- [46] Charles R. Johnson. *Matrix Theory and Applications*. en. Google-Books-ID: KHRHCQAAQBAJ. American Mathematical Soc., 1990. ISBN: 978-0-8218-0154-3.
- [47] Garth S. Jowett and Victoria O’Donnell. *Propaganda & Persuasion*. en. Google-Books-ID: ThhcD-wAAQBAJ. SAGE Publications, Aug. 2018. ISBN: 978-1-5063-7135-1.
- [48] Dawid Jurkiewicz et al. “ApplicaAI at SemEval-2020 Task 11: On RoBERTa-CRF, Span CLS and Whether Self-Training Helps Them”. en. In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, 2020, pp. 1415–1424. DOI: 10.18653/v1/2020.semeval-1.187. URL: <https://aclanthology.org/2020.semeval-1.187> (visited on 11/11/2024).
- [49] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. en. arXiv:1412.6980 [cs]. Jan. 2017. DOI: 10.48550/arXiv.1412.6980. URL: <http://arxiv.org/abs/1412.6980> (visited on 04/16/2025).
- [50] Sandeep Kumar. *Comparison of Sigmoid, Tanh and ReLU Activation Functions*. en-US. Aug. 2020. URL: <https://www.aitude.com/comparison-of-sigmoid-tanh-and-relu-activation-functions/> (visited on 04/16/2025).
- [51] Kyumin Lee, Steve Webb, and Hancheng Ge. “The Dark Side of Micro-Task Marketplaces: Characterizing Fiverr and Automatically Detecting Crowdurfing”. en. In: *Proceedings of the International AAAI Conference on Web and Social Media* 8.1 (May 2014), pp. 275–284. ISSN: 2334-0770, 2162-3449. DOI: 10.1609/icwsm.v8i1.14528. URL: <https://ojs.aaai.org/index.php/ICWSM/article/view/14528> (visited on 04/09/2025).
- [52] Dailin Li et al. “914isthebest at SemEval-2024 Task 4: CoT-based Data Augmentation Strategy for Persuasion Techniques Detection”. en. In: () .
- [53] Jing Li et al. “A Survey on Deep Learning for Named Entity Recognition”. In: *IEEE Transactions on Knowledge and Data Engineering* 34.1 (Jan. 2022), pp. 50–70. ISSN: 1558-2191. DOI: 10.1109/TKDE.2020.2981314. URL: <https://ieeexplore.ieee.org/abstract/document/9039685> (visited on 04/28/2025).

- [54] Irina Lock and Ramona Ludolph. “Organizational propaganda on the Internet: A systematic review”. en. In: *Public Relations Inquiry* 9.1 (Jan. 2020), pp. 103–127. ISSN: 2046-147X, 2046-1488. DOI: 10 . 1177 / 2046147X19870844. URL: <https://journals.sagepub.com/doi/10.1177/2046147X19870844> (visited on 04/02/2025).
- [55] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. en. arXiv:1711.05101 [cs]. Jan. 2019. DOI: 10 . 48550/arXiv.1711.05101. URL: <http://arxiv.org/abs/1711.05101> (visited on 04/16/2025).
- [56] Angel R. Martinez. “Part-of-speech tagging”. en. In: *WIREs Computational Statistics* 4.1 (2012). eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/wics.195>, pp. 107–113. ISSN: 1939-0068. DOI: 10 . 1002/wics.195. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.195> (visited on 04/28/2025).
- [57] Giovanni Da San Martino et al. *A Survey on Computational Propaganda Detection*. en. arXiv:2007.08024 [cs]. July 2020. URL: <http://arxiv.org/abs/2007.08024> (visited on 10/07/2024).
- [58] Yann Mathet, Antoine Widlöcher, and Jean-Philippe Métivier. “The Unified and Holistic Method Gamma () for Inter-Annotator Agreement Measure and Alignment”. en. In: *Computational Linguistics* 41.3 (Sept. 2015), pp. 437–479. ISSN: 0891-2017, 1530-9312. DOI: 10 . 1162/COLI_a_00227. URL: <https://direct.mit.edu/coli/article/41/3/437-479/1524> (visited on 04/10/2025).
- [59] Muskaan Maurya. *Name Entity Recognition and various tagging schemes*. en. Feb. 2023. URL: https://medium.com/@muskaan_maurya06/name-entity-recognition-and-various-tagging-schemes-533f2ac99f52 (visited on 04/30/2025).
- [60] Michele Mazza et al. *RTbust: Exploiting Temporal Patterns for Botnet Detection on Twitter*. en. arXiv:1902.04506 [cs]. Feb. 2019. DOI: 10 . 48550/arXiv.1902.04506. URL: <http://arxiv.org/abs/1902.04506> (visited on 04/28/2025).
- [61] *Media Bias/Fact Check - Search and Learn the Bias of News Media*. URL: <https://mediabiasfactcheck.com/> (visited on 04/28/2025).
- [62] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. en. arXiv:1301.3781 [cs]. Sept. 2013. DOI: 10 . 48550/arXiv.1301.3781. URL: <http://arxiv.org/abs/1301.3781> (visited on 04/24/2025).
- [63] David Miller and William Dinan. *A century of spin: How public relations became the cutting edge of corporate power*. en. London, United Kingdom: Pluto Press, Jan. 2008. ISBN: 978-0-7453-2688-7. URL: <https://strathprints.strath.ac.uk/27825/> (visited on 04/29/2025).
- [64] Kevin Moloney. *Rethinking Public Relations: PR Propaganda and Democracy*. 2nd ed. London: Routledge, Apr. 2006. ISBN: 978-0-203-03059-2. DOI: 10 . 4324/9780203030592.
- [65] Gaku Morio et al. “Hitachi at SemEval-2020 Task 11: An Empirical Study of Pre-Trained Transformer Family for Propaganda Detection”. en. In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, 2020, pp. 1739–1748. DOI: 10 . 18653/v1/2020.semeval-1.228. URL: <https://aclanthology.org/2020.semeval-1.228> (visited on 10/07/2024).
- [66] *nela-features: A package to compute text features for news veracity*. URL: <https://github.com/BenjaminDHorne/NELAFeatures> (visited on 05/14/2025).
- [67] *NLLoss — PyTorch 2.7 documentation*. URL: <https://docs.pytorch.org/docs/stable/generated/torch.nn.NLLLoss.html#torch.nn.NLLLoss> (visited on 05/14/2025).
- [68] Joseph S. Nye. “Soft Power”. In: *Foreign Policy* 80 (1990). Publisher: Washingtonpost.Newsweek Interactive, LLC, pp. 153–171. ISSN: 0015-7228. DOI: 10 . 2307 / 1148580. URL: <https://www.jstor.org/stable/1148580> (visited on 05/12/2025).
- [69] *Ollama*. URL: <https://ollama.com> (visited on 04/09/2025).
- [70] *OpenWebText2*. URL: <https://openwebtext2.readthedocs.io/en/latest/> (visited on 05/14/2025).
- [71] Jakub Piskorski et al. “SemEval-2023 Task 3: Detecting the Category, the Framing, and the Persuasion Techniques in Online News in a Multi-lingual Setup”. en. In: *Proceedings of the The 17th International Workshop on Semantic Evaluation (SemEval-2023)*. Toronto, Canada: Association for Computational Linguistics, 2023, pp. 2343–2361. DOI: 10 . 18653/v1/2023.semeval-1.317. URL: <https://aclanthology.org/2023.semeval-1.317> (visited on 10/24/2024).
- [72] *Poisoning the well*. en. Page Version ID: 1278064182. Feb. 2025. URL: https://en.wikipedia.org/w/index.php?title=Poisoning_the_well&oldid=1278064182 (visited on 04/08/2025).
- [73] *PyTorch*. en. URL: <https://pytorch.org/> (visited on 04/09/2025).

- [74] Alec Radford et al. “Language Models are Unsupervised Multitask Learners”. en. In: ().
- [75] Colin Raffel et al. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. en. arXiv:1910.10683 [cs]. Sept. 2023. DOI: 10.48550/arXiv.1910.10683. URL: <http://arxiv.org/abs/1910.10683> (visited on 04/24/2025).
- [76] Hannah Rashkin et al. “Truth of Varying Shades: Analyzing Language in Fake News and Political Fact-Checking”. en. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, 2017, pp. 2931–2937. DOI: 10.18653/v1/D17-1317. URL: <http://aclweb.org/anthology/D17-1317> (visited on 10/07/2024).
- [77] Espen Geelmuyden Rød and Nils B Weidmann. “Empowering activists or autocrats? The Internet in authoritarian regimes”. en. In: *Journal of Peace Research* 52.3 (May 2015), pp. 338–351. ISSN: 0022-3433, 1460-3578. DOI: 10.1177/0022343314555782. URL: <https://journals.sagepub.com/doi/10.1177/0022343314555782> (visited on 04/08/2025).
- [78] *SemEval*. en-US. URL: <https://semeval.github.io/> (visited on 05/12/2025).
- [79] *Semiotics for Beginners: Signs*. URL: <https://www.cs.princeton.edu/~chazelle/courses/BIB/semino2.htm> (visited on 05/12/2025).
- [80] David Shambaugh. “China’s Propaganda System: Institutions, Processes and Efficacy”. en. In: ().
- [81] Paramansh Singh. *paramansh/propaganda_detection*. original-date: 2020-01-11T18:09:31Z. Feb. 2025. URL: https://github.com/paramansh/propaganda_detection (visited on 04/30/2025).
- [82] *spaCy · Industrial-strength Natural Language Processing in Python*. en. URL: <https://spacy.io/> (visited on 04/14/2025).
- [83] *Spearman’s rank correlation coefficient*. en. Page Version ID: 1284974441. Apr. 2025. URL: https://en.wikipedia.org/w/index.php?title=Spearman%27s_rank_correlation_coefficient&oldid=1284974441 (visited on 05/14/2025).
- [84] *Stanford CS 224N — Natural Language Processing with Deep Learning*. URL: <https://web.stanford.edu/class/cs224n/> (visited on 04/07/2025).
- [85] Richard S Sutton and Andrew G Barto. “Reinforcement Learning: An Introduction”. en. In: ().
- [86] Philip M. Taylor. *Munitions of the mind: a history of propaganda from the ancient world to the present era*. English. Third. Book, Whole. Manchester;New York; Manchester University Press, 2013. ISBN: 9781847790927;1847790925; URL: <https://go.exlibris.link/wOBmJSdZ>.
- [87] *Transformers*. URL: <https://huggingface.co/docs/transformers/en/index> (visited on 04/29/2025).
- [88] *U.S. News & World Report: News, Rankings and Analysis on Politics, Education, Healthcare and More*. en. URL: <https://www.usnews.com> (visited on 04/28/2025).
- [89] *Understanding LSTM Networks – colah’s blog*. URL: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/> (visited on 04/16/2025).
- [90] Université Côte d’Azur, Inria, CNRS, I3S, France et al. ““Don’t discuss”: Investigating Semantic and Argumentative Features for Supervised Propagandist Message Detection and Classification”. en. In: *Proceedings of the Conference Recent Advances in Natural Language Processing - Deep Learning for Natural Language Processing Methods and Applications*. INCOMA Ltd. Shoumen, BULGARIA, 2021, pp. 1498–1507. ISBN: 978-954-452-072-4. DOI: 10.26615/978-954-452-072-4_168. URL: <https://acl-bg.org/proceedings/2021/RANLP%202021/pdf/2021.ranlp-1.168.pdf> (visited on 11/03/2024).
- [91] Ashish Vaswani et al. “Attention is All you Need”. en. In: ().
- [92] Alex Wang et al. *GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding*. en. arXiv:1804.07461 [cs]. Feb. 2019. DOI: 10.48550/arXiv.1804.07461. URL: <http://arxiv.org/abs/1804.07461> (visited on 04/24/2025).
- [93] Alex Wang et al. *SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems*. en. arXiv:1905.00537 [cs]. Feb. 2020. DOI: 10.48550/arXiv.1905.00537. URL: <http://arxiv.org/abs/1905.00537> (visited on 04/24/2025).
- [94] *Welcome to Python.org*. en. Apr. 2025. URL: <https://www.python.org/> (visited on 04/09/2025).
- [95] Anthony Weston. *A Rulebook for Arguments*. en. Google-Books-ID: XhVNDwAAQBAJ. Hackett Publishing, Feb. 2018. ISBN: 978-1-62466-655-1.

- [96] *What is Gradient Descent?* — IBM. en. Oct. 2021. URL: <https://www.ibm.com/think/topics/gradient-descent> (visited on 04/16/2025).
- [97] Samuel C Woolley. “Political Communication, Computational Propaganda, and Autonomous Agents - Introduction”. en. In: ().
- [98] *WordNet*. en. URL: <https://wordnet.princeton.edu/homepage> (visited on 04/24/2025).
- [99] Junchao Wu et al. “A Survey on LLM-Generated Text Detection: Necessity, Methods, and Future Directions”. In: *Computational Linguistics* 51.1 (Mar. 2025), pp. 275–338. ISSN: 0891-2017. DOI: 10.1162/coli_a_00549. URL: https://doi.org/10.1162/coli_a_00549 (visited on 04/28/2025).
- [100] Kai-Cheng Yang et al. “Arming the public with artificial intelligence to counter social bots”. en. In: *Human Behavior and Emerging Technologies* 1.1 (Jan. 2019), pp. 48–61. ISSN: 2578-1863, 2578-1863. DOI: 10.1002/hbe2.115. URL: <https://onlinelibrary.wiley.com/doi/10.1002/hbe2.115> (visited on 04/02/2025).
- [101] Sarita Yardi et al. “Detecting spam in a Twitter network”. en. In: *First Monday* (Dec. 2009). ISSN: 1396-0466. DOI: 10.5210/fm.v15i1.2793. URL: <https://journals.uic.edu/ojs/index.php/fm/article/view/2793> (visited on 04/02/2025).
- [102] *Zotero — Your personal research assistant*. URL: <https://www.zotero.org/> (visited on 04/07/2025).