HyperPT

Detection and Classification of Hyperpartisan News Articles

Mark Muscat

Supervised by Dr Joel Azzopardi

Co-supervised by Dr Colin Layfield

Department of Artificial Intelligence

Faculty of ICT

University of Malta

August, 2020

A dissertation submitted in partial fulfilment of the requirements for the degree of M.Sc. in Artificial Intelligence.

L-Università
ta' Malta

WWW.UM.EDU.MT

First edition, February 12, 2021

To Fellow Researchers

For embarking on such journeys of self-betterment and contribution to the Research Community.

Acknowledgements

Seeing this project through was not an easy task, yet with the continuous help and support provided by mentors, friends and close relatives, I find myself at its conclusions.

My deepest gratitude goes to my supervisor, Dr. Joel Azzopardi and my co-supervisor Dr. Colin Layfield, for their steady counselling and feedback guiding me all the way through from the project's inception to its end.

A heartfelt thanks also goes to close friends with whom I embarked on this journey and shared experiences along the way. Thanks also goes to my family, for their constant support and help in managing such a task.

Last but definitely not least, I am infinitely grateful to Maria, my life companion to whom I dedicate this accomplishment for being there with me every step of the way, offering love, wisdom, and encouragement.

Abstract

The modern hyper-connected world brings with it an unprecedented rise in fake and hyperpartisan news, with anyone connected online harnessing the power of producing such fabricated information. Hyperpartisan news can be defined as extremely one-sided or biased news towards or against an entity. It differs from fake news by often exaggerating and sensationalising real-life events. With the spread of such malicious information, the otherwise subjective opinion of vulnerable consumers is compromised, twisted and possibly manipulated by some ulterior agenda - resulting in unprecedented and damaging outcomes as already seen in now world-wide known incidents.

We hence give our contribution to addressing this issue by introducing HyperPT, a classification system for the automatic detection of hyperpartisan news articles. Throughout this study we experiment with a number of data representations, classification algorithms and external article features with the aim of creating an accurate and reliable classification system. In doing so gaining further insight into the nature of the hyperpartisan news article.

From our experiments we conclude on an SVM-based classification system working on article features represented as deep contextualised ELMo embeddings. Moreover, we test the addition of sentiment within the classification while also experimenting with different news article lengths. Explainability A.I. is used to interpret the model's decision-making and determine the influence of the article features on the classification. Finally we compare our system with the current state-of-the-art, achieving a mean accuracy score of 0.8220 to the other's 0.8404. In doing so we hence present an alternative system for the classification of hyperpartisan news articles.

Contents

1 Introduction **1**

1.1 Motivation . . . 1

1.2 Proposed Solution . . . 3

1.3 Aims and Objectives . . . 4

1.4 Research Contributions . . . 6

1.5 Document Structure . . . 7

2 Background and Literature Review **9**

2.1 Hyperpartisan News Article Dataset . . . 9

2.1.1 PAN SemEval Hyperpartisan News Detection . . . 10

2.1.2 PAN SemEval Hyperpartisan News Dataset . . . 10

2.2 Feature Preprocessing & Representation . . . 13

2.2.1 Data Preprocessing for Hyperpartisan News Articles . . . 14

2.2.2 Feature Representation for Hyperpartisan News Articles . . . 16

2.3 Classification Approaches . . . 21

2.3.1 Traditional Approaches . . . 21

2.3.2 Deep Learning Approaches . . . 24

2.4 Sentiment Analysis . . . 28

2.4.1 Sentiment Analysis - Overview . . . 28

2.4.2 VADER for Sentiment Analysis . . . 29

2.5 Explainable A.I. . . . 30

2.5.1 Sensitivity Analysis . . . 31

2.5.2 Layerwise Relevance Propagation . . . 31

2.6 Evaluation Criteria . . . 33

2.7 Related Work . . . 35

2.7.1 Detection of Fake News 35
2.7.2 Detection of Hyperpartisan News 37
2.7.3 Sentiment in Hyperpartisan News Articles 42
2.7.4 Model Interpretability and Saliency of Features 43
2.8 Summary 44

3 System Methodology 47

3.1 System Overview 47
3.2 Data Loading, Preprocessing and Representation 50
3.2.1 Data Preprocessing 51
3.2.2 Feature Representation 51
3.3 Classification Approaches 53
3.3.1 Traditional ML Approaches 53
3.3.2 Deep Learning Approaches 54
3.3.3 System Implementation and Hyperparameter Tuning 56
3.3.4 Adapting Feature Representations to the Classifiers 56
3.4 External Features 57
3.4.1 Sentiment Features for Hyperpartisan News Classification 57
3.4.2 Explainable A.I. for Model Interpretation and Feature Saliency . . 59
3.5 Summary 61

4 System Evaluation and Discussion 63

4.1 Data Preprocessing and Representation 64
4.1.1 Feature Representation for Hyperpartisan News 65
4.1.2 Data Preprocessing for Hyperpartisan News 70
4.1.3 Feature Representation and Data Preprocessing - Discussion 75
4.2 Classifier 75
4.2.1 Hyperparameter Tuning for Hyperpartisan News Classification . . 76
4.2.2 Evaluating HyperPT Classifier performance 81
4.2.3 Classifier for Hyperpartisan News Classification - Discussion . . . 83
4.3 External Features 84
4.3.1 Sentiment as a Feature for Hyperpartisan News 84
4.3.2 Title and Body of a Hyperpartisan News Article 89
4.3.3 Model Explainability and Saliency of Hyperpartisan Features . . . 92
4.3.4 External Features of Hyperpartisan News - Discussion 100
4.4 HyperPT against the State-of-the-Art 100
4.4.1 The State-of-the-Art in Hyperpartisan News Classification 101

4.4.2 HyperPT vs the State-of-the-Art - Discussion 101
4.5 Summary 103

5 Conclusions 105
5.1 Achieved Aims and Objectives 106
5.1.1 Features of a Hyperpartisan News Article 106
5.1.2 Sentiment of a Hyperpartisan News Article 106
5.1.3 Minimum length of text for an Article to be Hyperpartisan 107
5.1.4 Classifier for Hyperpartisan News Articles 107
5.1.5 Interpretation of the Classifier 107
5.2 Critique and Limitations 108
5.3 Future Work 108
5.4 Final Remarks 110

References 111

List of Figures

1.1 A sample hyperpartisan news article 2
1.2 A sample non-hyperpartisan news article 2

2.1 A sample hyperpartisan-flagged news article in XML form. 11
2.2 A sample article title with punctuation removed. 15
2.3 Stemming using Porter′s and Snowball 15
2.4 Basic architecture of a Word2Vec CBOW 19
2.5 SVM Hyperplanes 23
2.6 A sentence classification using CNN 25
2.7 Basic architecture of a biRNN 27
2.8 A.I. interpretability using SA and LRP 33
2.9 The biLSTM Self-Attention Mechanism 44

3.1 The HyperPT Design Process 48

4.1 Performance of Data Representation Techniques 69
4.2 Performance of Embedded Sentiment Features 88
4.3 Evaluation of LRP on the SVM Classifier 93
4.4 Evaluation of LRP on the CNN Classifier 94
4.5 A hyperpartisan news article classified using SVM and examined using LRP . 95
4.6 A neutral news article classified using SVM and examined using LRP 96
4.7 A hyperpartisan news article classified using CNN and examined using LRP 97
4.8 A neutral news article classified using CNN and examined using LRP 97

List of Tables

2.1 Sample NLTK English Stopwords 14

3.1 Article and Sentence level Sentiment 59

4.1 Accuracy scores for individual Data Preprocessing using SVM. 66
4.2 Accuracy scores for individual Data Preprocessing using RF. 67
4.3 Accuracy scores for individual Data Preprocessing using CNN. 68
4.4 Accuracy scores with ELMo embedded features 71
4.5 Accuracy results for aggregate Data Preprocessing techniques. 73
4.6 SVM classifier hyperparameters tested using Grid-Search. 77
4.7 RF classifier hyperparameters tested using Grid-Search. 78
4.8 CNN classifier hyperparameters tested using Grid-Search. 80
4.9 Mean and highest classifier accuracy and F1 results. 82
4.10 Accuracy and F1 Scores using different sentiment embedding techniques . . . 86
4.11 Accuracy and F1 Scores on article title and body 90
4.12 Most salient hyperpartisan and neutral words 99
4.13 Mean and highest accuracy scores for HyperPT vs. State-of-the-Art 102

List of Abbreviations

A.I. Artificial Intelligence

BERT Bidirectional Encoder Representations from Transformers

BoW Bag-of-Words

CNN Convolutional Neural Network

DL Deep Learning

ELMo Embeddings from Language Models

GloVe Global Vectors

LR Logistic Regression

LRP Layerwise Relevance Propagation

LSTM Long Short-Term Memory

ML Machine Learning

NLP Natural Language Processing

NLTK Natural Language Toolkit

POS-Tags Part-of-Speech Tags

RBF Radial Basis Function

RF Random Forest

RNN Recurrent Neural Network

SA Sensitivity Analysis

SVM Support Vector Machine

TF Term Frequency

TF-IDF Term Frequency - Inverse Document Frequency

VADER Valence Aware Dictionary for sEntiment Reasoning

XAI Explainable A.I.

1

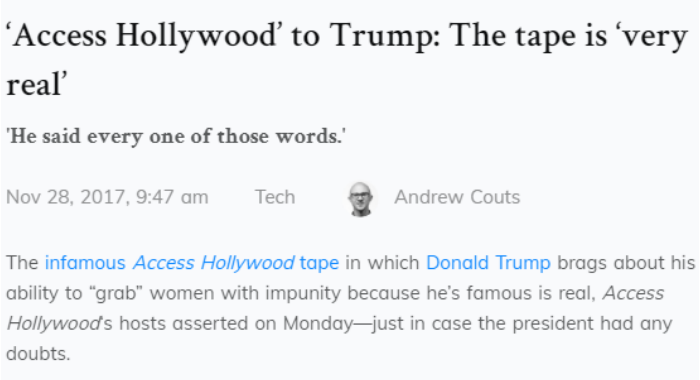
Introduction

During the 2016 United States elections, a data analytics company by the name of Cambridge Analytica is alleged to have purposely fabricated and spread false and targeted news with the aim of altering the outcome of the elections and possibly also the resulting Presidency [Berghel (2018)]. The firm is alleged to have exploited a vulnerability in a Facebook App allowing them to not only gather data on the users using the application, but also on all of their Facebook friends. The personal profiles of around 50 million people were exposed and targeted by specific, biased and hyperpartisan advertisements - personalised towards the different classes of profiles mined from this breach, potentially effecting the people's perception and voting at a mass scale throughout the elections [Berghel (2018); Cadwalladr and Graham-Harrison (2018a,b)].

1.1 | Motivation

Hyperpartisan news is defined as extremely one-sided, or biased news [Potthast et al. (2018)]. It is often pushed by a hidden agenda towards or against a specific entity or group of entities. Hyperpartisan news typically makes use of overly dramatic headlines and inflammatory wording with the aim of quickly capturing the reader's attention. News is reported with a degree of bias, with the author frequently including opinionated commentary on the reported material.

Different to fake news - which can be considered as containing a degree of fabricated untruths, hyperpartisan news often reports on actual, authentic events, although in a biased way. Despite this, both fake and hyperpartisan news articles typically contain the use of inflammatory and sensationalised vocabulary [Potthast et al. (2018)]. Fact-checking, a useful tool for the evaluation of fake news, is not a potent solution for hyperpartisan news, which due to the exaggerated reporting of real events, makes the detection of such news even more challenging. Two snippets, one of a hyperpartisan, and the other of a non-hyperpartisan news articles can be respectively examined in Figure 1.1 and Figure 1.2.



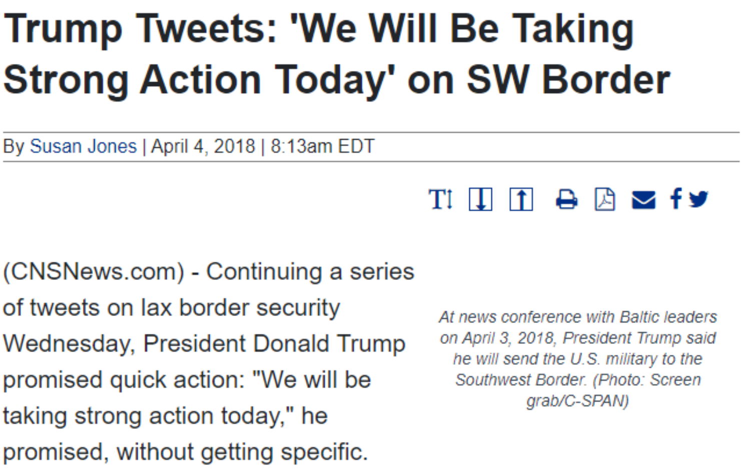
'Access Hollywood' to Trump: The tape is 'very real'

'He said every one of those words.'

Nov 28, 2017, 9:47 am Tech Andrew Couts

The infamous *Access Hollywood* tape in which Donald Trump brags about his ability to "grab" women with impunity because he's famous is real, *Access Hollywood*'s hosts asserted on Monday—just in case the president had any doubts.

Figure 1.1: A sample hyperpartisan news article[1]- labelled through crowd-sourcing.



Trump Tweets: 'We Will Be Taking Strong Action Today' on SW Border

By Susan Jones | April 4, 2018 | 8:13am EDT

(CNSNews.com) - Continuing a series of tweets on lax border security Wednesday, President Donald Trump promised quick action: "We will be taking strong action today," he promised, without getting specific.

At news conference with Baltic leaders on April 3, 2018, President Trump said he will send the U.S. military to the Southwest Border. (Photo: Screen grab/C-SPAN)

Figure 1.2: A sample non-hyperpartisan (neutral) news article[2]- labelled through crowd-sourcing.

In Figure 1.1, strong words such as *infamous, brags, impunity* are used, while unnecessary opinionated expressions like *just in case the president had any doubts* are also present. Moreover, the then president of the United States, Donald Trump, is referred to simply by his name. This contrasts to the concise approach employed in Figure 1.2, where both

[1] ***'Access Hollywood' to Trump: The tape is 'very real'*** - `www.dailydot.com` [Last Accessed: 07-2020]

[2] ***Trump Tweets: 'We Will Be Taking Strong Action Today' on SW Border*** - `www.cnsnews.com` [Last Accessed: 07-2020]

the headline and the article body are clear on their message, and the same individual is addressed by his full title (*President Donald Trump*).

The emergence and widespread distribution of hyperpartisan news inspires us to explore this domain from the lens of A.I., putting to use the power of Machine Learning (ML) [Michie et al. (1994)] at attempting to provide a pragmatic and efficient tool against the spread of such malicious information. Without these tools the sharing of hyperpartisan news goes uncontrolled, quickly overshadowing genuine, neutral news with its dramatic and sensationalised nature. Moreover, readers themselves often share such news with their peers before actively checking for authenticity, further speeding up the sharing process [Tambuscio et al. (2015)].

One finds notable progress already made in addressing both fake news and hyperpartisan news [Kiesel et al. (2019); Potthast et al. (2018)]. In the domain of fake news we see approaches tackling the spread, stylistic writing and content within, with the hopes of detecting such news early on, hindering its tendency to quickly spread and affect consumers online. Being arguably less known than fake news, we see this as an opportunity to pitch in our contribution in detecting hyperpartisan news - building on promising published work with the idea of further improving on the state-of-the-art and expanding on existing research. In doing so, we aim at not only introducing an efficient and reliable hyperpartisan news detection system, but also at examining the very nature of the hyperpartisan news article, with the hopes that this would inspire us along with fellow researchers, to extend on the existing research.

1.2 | Proposed Solution

In detecting hyperpartisan news articles, we feel that due to the often dramatic and sensationalised nature of such articles, the content within the article and its style of writing is the best medium to analyse. We hence form this challenge as a Machine Learning (ML) classification problem, where we represent the article data features as numerical vectors which are then passed to a classification algorithm, in turn giving us a corresponding prediction label defining whether the inputted document is hyperpartisan or neutral.

Being the direct medium upon which the classification process is performed, the features within a hyperpartisan news article are crucial to an accurate and reliable classification. One must first sufficiently clean the dataset from any noise and inconsistencies before classification takes place. Moreover, representation techniques converting the otherwise textual data being passed to the classifier are themselves a crucially important step throughout the whole classification process. As we see further on in Chapter 4, both data preprocessing and the corresponding data representations are pivotal to the overall performance.

Expanding further on the features making up the news articles, we feel that direct interpretation of the classifiers' decision-making through explainability algorithms may allow us to determine precisely the logic responsible for the classifier's behaviour leading up to such a classification. In other words, we define the article features contributing, or opposing, to the resulting classification label.

Thorough experimentation is considered, testing different lengths of hyperpartisan news articles along with the inclusion and exclusion of the article title. Such tests are performed with the intention of determining whether short texts such as tweets or simply article headlines can also be hyperpartisan, and whether the full length of a typical hyperpartisan news article would be necessary for reliable detection of such content. In doing so, one would be more knowledgeable as to which article lengths tend to be more prone to hyperpartisanship, and whether simpler forms of text are also vulnerable to such malicious information.

Furthermore, the sensationalised writing typically associated with hyperpartisan news raises interest as to the potential role sentiment may play in the detection of such news. Basing on the observations of similar systems [Kiesel et al. (2019)] that heavy sentimental elements tend to be present within such texts, we take into consideration the detection of sentiment within hyperpartisan articles and find a way of testing them integrated along with the rest of the detection process.

Finally, we evaluate three promising classification algorithms (Random Forest [RF], Support Vector Machine [SVM] and Convolutional Neural Network [CNN]) currently in use in both similar systems on the same area of research and imported from other disciplines. We compare the performance of these algorithms with that of one another along with similar published work in order to settle on the best performing classification model, thereby developing and evaluating our hyperpartisan news article classification system.

1.3 | Aims and Objectives

Through our proposed solution discussed in Section 1.2, we hence intend to create a pragmatic and reliable hyperpartisan news detection system. In doing so, we aim at gaining insight into the nature of the hyperpartisan article itself, with the hopes that this may further aid us and future research in improving the work conducted.

We hence divide our project into five individual yet complementary objectives which we plan to address in order to achieve our main goal:

1. *Features of a Hyperpartisan news article*: Discover the most salient features indicating that a news article is hyperpartisan. Some features may prove to have more influence on the classification outcome than others. In analysing the saliency of each feature, we could determine which features are the most important.

2. *Sentiment of a Hyperpartisan news article*: Experiment with different sentiment integration techniques and examine their effects on the detection of hyperpartisan news articles.

3. *Minimum length of text for an article to be Hyperpartisan*: Determine the least amount of textual data that is required for an article to be classified as hyperpartisan. Does the title suffice, or does the article body play a crucial part as well? If so, what is the ideal body length?

4. *Classifier for Hyperpartisan news articles*: Harnessing the knowledge acquired from the previous points, research and develop the best performing classification system for the detection of hyperpartisan news articles.

5. *Interpretation of the Classifier*: Use the capabilities brought forward by Explainable A.I. - specialised methods capable of analysing the classifier and its logic, to interpret the model's decision-making behind its classifications. In doing so one could then determine the model's generalisation capabilities and its reliability.

With the data features and preprocessing stages playing such a pivotal role within the classification process, we feel that the attention given to the typical characteristics of hyperpartisan articles (as detailed in the first three objectives) is of benefit to both our study and future work. We therefore take into consideration not only the article textual features, but also the sentiment, length of the article body, and role of the article title - with the hopes that these would provide further insight into the importance of such attributes.

Moreover, we evaluate three classification algorithms, the RF, SVM, and CNN. These three approaches are chosen from research conducted on related work purposely for the classification of hyperpartisan news articles. In evaluating these classifiers we monitor their accuracy and performance while also using specialised approaches for the direct interpretation of the model's behaviour. In doing so we aim at determining whether the model's approach is as intended, while also attempting at predicting its expected performance with the classification of new data samples.

1.4 | Research Contributions

In tackling the objectives highlighted in Section 1.3, we introduce HyperPT - our concept for a hyperpartisan news article detection system. We aim at improving on already published research, focusing on the features within hyperpartisan news articles and employing ML techniques to create a reliable classification system. Such a system would need to have the capability of assessing whether an article is of a neutral or hyperpartisan nature. In building the HyperPT system, we attempt at tackling two individual aspects simultaneously; the proposal and building of an accurate and reliable classification system, and the discovery of further knowledge of the typical nature of hyperpartisan news content.

In conducting our study, we research reputable classification solutions within the ML discipline and moreover in the detection of hyperpartisan news content. Following our research, we choose the Support Vector Machine (SVM), Random Forest (RF) and Convolutional Neural Network (CNN) as our three candidate classifiers. Tests are performed with a number of data preprocessing and various feature representation techniques - both those of a traditional nature (Bag-of-Words [BoW], Term Frequency - Inverse Document Frequency [TF-IDF], Part-of-Speech Taggings [POS-Tags]) and those more associated with modern Deep Learning approaches, namely word embeddings (Word2Vec, GloVe, ELMo, and BERT).

We compare the performance of the RF, SVM and CNN; 1) Finding the optimal feature representation and hyperparameter configuration for maximising the classification accuracy of each classifier, 2) Comparing and contrasting the classical ML classifiers with the more elaborate DL [LeCun et al. (2015)] classifiers, and finally 3) Choosing the best performing system among the three classifiers, after evaluating the techniques' performance and results acquired in the second step. Experiments conducted compel us to settle on the SVM classifier (Chapter 4), in doing so giving our thoughts on the reasons behind the clear performance superiority over the two other candidates.

Simultaneously with the experiments above, we explore both the hyperpartisan article textual features and external characteristics, namely sentiment, the article title and length of article body. Inspired by similar work and the sensationalised nature typically found within such articles, we particularly experiment with different incorporations of the sentiment within, integrating sentiment score and labels in several configurations among the internal features defining the original articles. With the unexpected hindrance to the system performance resulting from the addition of sentiment, we hypothesise on the underlying explanations.

Finally, we implement and evaluate a model explanation technique known as Layerwise Relevance Propagation (LRP) to interpret the decision-making of our classification models - particularly the SVM and the CNN. This in itself gives us two benefits; the first and most obvious is that it allows us to gauge the performance of the classifier, determining whether it is working as intended and if is capable of generalising to new data samples. Simultaneously however, this also allows us to explore the news article features and their saliency within the classification. Analysing the acquired saliency results, we notice a correlation between entities (namely individuals and events) addressed within the articles and the corresponding hyperpartisan labels.

1.5 | Document Structure

Throughout this document we describe our approach to implementing the HyperPT study into five chapters as follows:

1. **Introduction** - Throughout the Introduction, we have discussed the problem presented by fake and hyperpartisan news articles. We have proposed five main objectives which we aim to tackle, thereby not only presenting an accurate and reliable classification system, but also exploring further the nature of hyperpartisan news articles.

2. **Background and Literature Review** - Before delving into the proposed system, we first provide a concise background on algorithms and approaches which are in some way relevant to the project. This is followed by an elaborate review of related work, highlighting systems employing solutions for the detection of fake and hyperpartisan news articles.

3. **System Methodology** - The system methodology is presented and discussed, where we detail our approach, reasoning behind decisions taken, and physical implementation of the system. Visualised also as a flowchart, we discuss the three main components making up the HyperPT system, along with smaller subcomponents utilised ad hoc as preprocessing and runtime steps.

4. **System Evaluation and Discussion** - We evaluate the three proposed classification algorithms by comparing them altogether. Tests are performed using a range of different data preprocessing and representation techniques, including the addition of sentiment features. Finally, the LRP explainability algorithm is evaluated, before utilised for determining the article feature saliency.

5. **Conclusions** - Throughout this conclusive chapter, our final remarks on the HyperPT project are given. We discuss the implemented system, its limitations and potential future work - which would undoubtedly further extend and improve on the work conducted here.

2

Background and Literature Review

Having discussed the reality behind hyperpartisan news articles and the problem they entail, we introduced our system, HyperPT, for the detection of such malicious content. Before delving further into the system and its components, we now give a concise background on technologies and approaches which are either an inspiration to, or are directly used within HyperPT.

Throughout this chapter, we first examine the SemEval Hyperpartisan News Articles Dataset (Section 2.1), which is the main and only dataset upon which the HyperPT system is built. We proceed by discussing feature preprocessing and representation methods in Section 2.2. A selection of classifiers is analysed in Section 2.3, thereby completing the baseline classification system.

The subsequent two sections; Section 2.4 and Section 2.5, would then respectively focus on sentiment features and model explainability. This is followed by an overview of the project's evaluation criteria, discussed in Section 2.6. Finally, in Section 2.7 we present a review of related systems employing the approaches detailed previously (or similar techniques). With this we conclude the chapter and proceed to the design and implementation process, detailed in Chapter 3.

2.1 | Hyperpartisan News Article Dataset

Before delving into the algorithms themselves, we first discuss the dataset supplying us with both hyperpartisan and neutral labelled articles. In Section 2.1.1 we discuss the PAN SemEval Hyperpartisan News Detection [Kiesel et al. (2019)] competition, a challenge through which the organisers introduce the Hyperpartisan News dataset, while kickstarting research on the topic of hyperpartisan news detection. The corresponding dataset is thoroughly discussed right after in Section 2.1.2.

2.1.1 | PAN SemEval Hyperpartisan News Detection

PAN[1] is a series of scientific events and a community of shared knowledge on digital text and stylometry. Among the range of hosted competitions, one finds the PAN SemEval Hyperpartisan News Detection competition[2], held in 2019.

The objective of the task is simple, given a dataset consisting of labelled mainstream and hyperpartisan news articles, create a system which is able to efficiently and reliably distinguish between the two. In all, 42 teams took part, with Jiang et al. (2019), also known as team Bertha Von Suttner, achieving a classification accuracy of 0.84 and winning the SemEval Hyperpartisan News Detection competition.

2.1.2 | PAN SemEval Hyperpartisan News Dataset

The dataset[3] presented for the PAN SemEval Hyperpartisan News Detection is assembled by Kiesel et al. (2019). It is split into two parts, known as the By-Article collection and the By-Publisher collection.

Both collections within the dataset contain articles of both left-wing and right-wing agendas, yet since both agendas have been shown to share more stylistic similarities between them than with mainstream news [Potthast et al. (2018)], we refrain from taking into consideration any political sides, and focus our full attention on the binary classification of whether an article is hyperpartisan or neutral.

The By-Article collection consists of 1273 articles labelled through crowdsourcing on an article basis. In other words, each article is peer-reviewed by multiple individuals, with an overall agreement on whether it is of a hyperpartisan or neutral nature. Of these articles, solely 645 are made public (and used by external studies such as ours), with the rest being maintained privately for the evaluation of competing systems. Of these 645, 37% (238) are hyperpartisan, with the other 63% (407) of articles being mainstream.

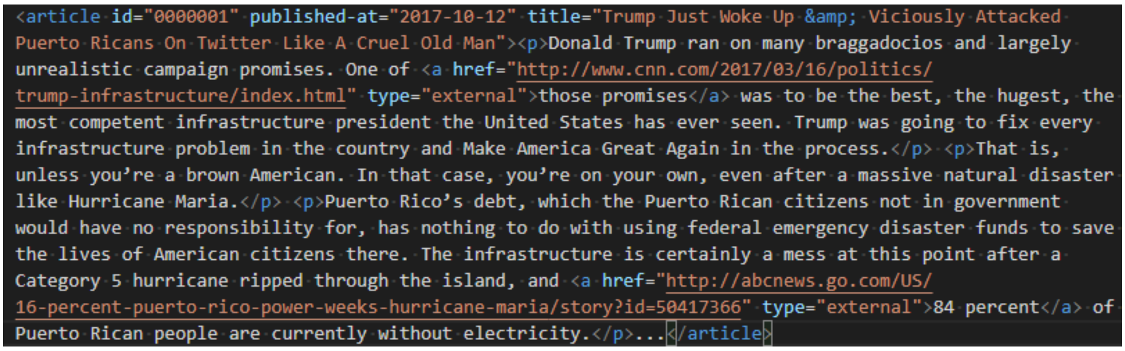
The By-Publisher collection on the other hand is significantly larger and consists of $754,000$ articles; $600,000$ of which are released as a public dataset, with the remaining being split into a public validation set ($150,000$) and a private evaluation set (4000). All of these sets consist of 50% hyperpartisan and 50% mainstream articles. Different to the By-Article collection, these articles are labelled by the overall bias label of their publishing source, as given by BuzzFeed journalists and MediaBiasFactCheck[4]. This

[1] ***PAN Scientific Events*** - `pan.webis.de` [Last Accessed: 07-2020]

[2] ***PAN SemEval Hyperpartisan News Detection (2019)*** - `pan.webis.de/semeval19` [Last Accessed: 07-2020]

[3] ***PAN SemEval Hyperpartisan News Dataset*** - `www.zenodo.org` [Last Accessed: 07-2020]

[4] ***MediaBiasFactCheck*** - `www.mediabiasfactcheck.com` [Last Accessed: 07-2020]



```
<article id="0000001" published-at="2017-10-12" title="Trump Just Woke Up &amp; Viciously Attacked
Puerto Ricans On Twitter Like A Cruel Old Man"><p>Donald Trump ran on many braggadocios and largely
unrealistic campaign promises. One of <a href="http://www.cnn.com/2017/03/16/politics/
trump-infrastructure/index.html" type="external">those promises</a> was to be the best, the hugest, the
most competent infrastructure president the United States has ever seen. Trump was going to fix every
infrastructure problem in the country and Make America Great Again in the process.</p> <p>That is,
unless you’re a brown American. In that case, you’re on your own, even after a massive natural disaster
like Hurricane Maria.</p> <p>Puerto Rico’s debt, which the Puerto Rican citizens not in government
would have no responsibility for, has nothing to do with using federal emergency disaster funds to save
the lives of American citizens there. The infrastructure is certainly a mess at this point after a
Category 5 hurricane ripped through the island, and <a href="http://abcnews.go.com/US/
16-percent-puerto-rico-power-weeks-hurricane-maria/story?id=50417366" type="external">84 percent</a> of
Puerto Rican people are currently without electricity.</p>...</article>
```

Figure 2.1: A sample hyperpartisan-flagged news article[5]in XML form as supplied by Kiesel et al. (2019) for the SemEval Hyperpartisan News Detection competition and the public.

implies that a hyperpartisanship label is assigned to the publishing entity, with all of its published articles inheriting the same label.

The dataset is made available publicly following the SemEval event, and is downloadable on request. We acquired it in XML format (as shown in Figure 2.1), with plans to give our contribution to the field of hyperpartisan news detection, evaluating our system on the same grounds as other published systems. Each article in XML form consists of a unique article ID, article title, date of publication, article URL and the article body. A separate XML file known as the ground truth is provided, containing the corresponding hyperpartisan labels for each article. Each article label is of a Boolean nature, indicating whether the article is hyperpartisan (True) or neutral (False).

Inside the article body, one often finds links to other related webpages [Jiang et al. (2019); Ning et al. (2019)]. These are represented using the URL <*a href*> tag, and sit among the rest of the article textual features. Moreover, being directly scraped off webpages, other unwanted text such as advertisements is sometimes included with the article body. We observed that a distinguishing factor between these types of texts and the actual article body is the <*p*> tags, since article-relevant body text is typically found within these tags.

2.1.2.1 | Dataset Labelling for Classification

As reported by a number of participating teams [Alabdulkarim and Alhindi (2019); Hanawa et al. (2019); Palić et al. (2019); Yeh et al. (2019)], the By-Article collection within the dataset is notably more reliable in its labellings than the By-Publisher, resulting in a

[5] *Trump Just Woke Up & Viciously Attacked Puerto Ricans On Twitter Like A Cruel Old Man* - `www.bipartisanreport.com` [Last Accessed: 07-2020]

substantially better training dataset and better classifier performance. The authors suggest that this is since labelling performed on the By-Article collection is personalised to each article, while labelling on the By-Publisher collection is done as an aggregate process based on the typical bias of the publishing entity. This is despite the varying levels of hyperpartisanship across the published articles - since it is unlikely for every article to have the same consistent levels of hyperpartisan bias.

Poor or inconsistent labelling containing high degree of noise makes it more difficult for the classifier to train and generalise properly on the training data - as is apparent in research conducted by similar systems. Indeed one observes large performance discrepancies among the same classification systems when trained on the By-Publisher collection to when trained on the By-Article [Kiesel et al. (2019)]. Palić et al. (2019) train an SVM classifier on both collections, with the By-Article achieving accuracies upwards of 75% and the By-Publisher in the ranges of 58% to 62%. Similarly, one finds systems such as Jiang et al. (2019) and Isbister and Johansson (2019) which boast state-of-the-art performance when training on the By-Article collection, yet suffer significantly on the By-Publisher. Other studies choose to ignore the latter collection altogether [Kiesel et al. (2019)].

One finds various approaches in trying to address the limitations imposed by poor data labelling. Several systems use just segments of the By-Publisher collection in order to aid in the classification process. Hanawa et al. (2019) extract N-grams from the By-Publisher collection in order to assemble a phrase set to be used as features along with the primary classification on the By-Article collection. Similarly, Drissi et al. (2019) use the By-Publisher collection for further training following initial training on the By-Article collection.

Alternatively, similar systems attempt at de-noising and re-labelling the data points available within the dataset. Lee et al. (2019) use pseudo-labelling, a semi-supervised learning approach, to filter out noisy labels from within the By-Publisher collection by approximating new labels. In doing so, the authors extract a de-noised dataset of around $32,000$ articles. On Similar terms, Pérez-Almendros et al. (2019) train two SVMs and a BiLSTM on the By-Article collection, before applying them as a meta-classifier on the By-Publisher collection. In this way, articles having the corresponding label matching the classification are kept, while all others are discarded.

The newly emerging discipline of Explainable A.I. (XAI) [Goebel et al. (2018); Samek et al. (2017)] may provide further tools with which one is able to determine the quality of the labels and the levels of noise within a dataset. In case of having a dataset with a volume of noisy labels, as is reportedly the case for the By-Publisher article collection, XAI can be used to analyse the classifier's decision-making and determine whether the logic behind it is both correct and reliable [Gade et al. (2019); Goebel et al. (2018)]. In this way one could identify pivotal features within the classification, and fine-tune the classification system accordingly. Having a fine-tuned system, one could then compare the noisy dataset labels with those predicted, and replace the noisy labels with the newly predicted ones, thereby attempting to de-noise the dataset.

Being a novel subject within the area of A.I., we do not, at the time of writing, find any published work attempting to use this method for the de-noising of the SemEval Hyperpartisan By-Publisher article collection. Having said so, one does find a minority of similar systems [Amason et al. (2019); Zhang et al. (2019)] employing XAI for the identification of salient features, as discussed further in Section 2.7.4.

With a limited amount of time available for the project development, we employ solely the By-Article collection for the detection of hyperpartisan news articles. We decide against incorporating the By-Publisher collection as well since this would enlarge the already elaborate scale of the project. Having said so, we do think there is potential in considering, as future work, the de-noising and integration of the By-Publisher collection within the classification of hyperpartisan news articles.

With a concise overview of the Hyperpartisan dataset, in Section 2.2 we now provide an elaborate background on the techniques used in building the HyperPT study, discussing the methodology behind each approach.

2.2 | Preprocessing and Representation of Hyperpartisan News Article Features

The extraction, selection and preparation of data features is a crucial part of the process in addressing any ML problem - in our case; the prediction of hyperpartisan articles. The performance of any classification model strongly depends on the condition and quality of the features it is given. It is thereby important that data is cleaned and represented in such a way that is optimised and ideal for the classifier to work with.

In Section 2.2.1 we examine the preprocessing typically applied to textual features before textual classification tasks. In Section 2.2.2 we then examine traditional and DL feature representation approaches for textual data. Traditional systems include the widely used Bag-of-Words and TF-IDF approaches, among others, while coming to DL, we then open up on the concept of Word Embeddings.

2.2.1 | Data Preprocessing for Hyperpartisan News Articles

Before being effectively processed by a classification system, data features typically pass through a preprocessing layer, in order to be refined as much as possible. When working with textual features, data preprocessing and cleaning often entails:

- Removal of stopwords
- Removal of punctuation
- Lowercasing of features
- Reducing features to their stem or lemma

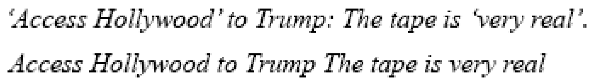
Stopword Removal: Stopwords are defined as words which are equally common in both documents relevant to a query and documents which are non-relevant [Wilbur and Sirotkin (1992)]. In other words, they are common features which do not typically give any relevant information on a specific subject, but are simply in the text to complete the lexical structure of the language. The Natural Language Toolkit (NLTK) [Bird et al. (2009); Loper and Bird (2002)] is an open source language toolkit which comes preconfigured with a list of 127 English stopwords (some of which are displayed in Table 2.1) which is often used to easily clarify which words to exclude from a given text snippet.

do	but	at
does	if	by
did	or	for
doing	because	with
a	as	about
an	until	against
the	while	between
and	of	into

Table 2.1: A sample of 24 stopwords within NLTK for the English Language.

Punctuation Removal: In tokenising the features of a given text, punctuation may also be removed [Ning et al. (2019); Zehe et al. (2019)]. Along with the removal of stopwords, this step further cleans the textual features, reducing them to individual words as showcased in Figure 2.3. Typically, one simply removes the punctuation characters within a text snippet, or replaces each oneċ with a white-space.

[6] *'Access Hollywood' to Trump: The tape is 'very real'* - www.dailydot.com [Last Accessed: 07-2020]



'Access Hollywood' to Trump: The tape is 'very real'.

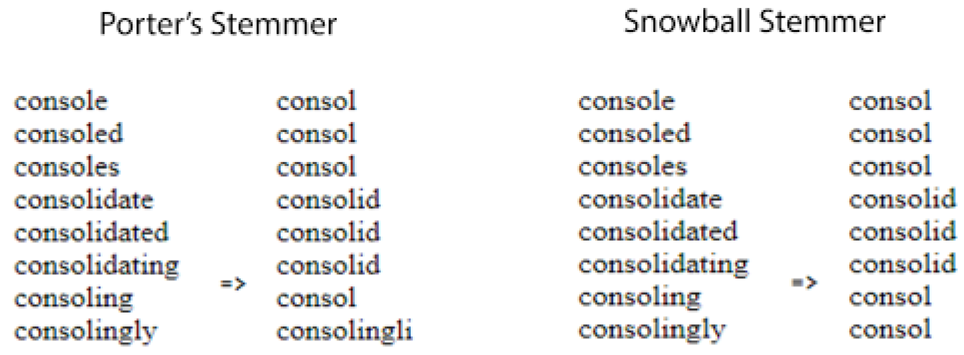
Access Hollywood to Trump The tape is very real

Figure 2.2: A sample hyperpartisan-flagged news article[6]title with and without punctuation.

Feature Lowercasing: One may also consider lowercasing all words such that any uppercased or capitalised words are 'normalised' to lowercase along with the rest of the corpus. We find systems such as Shaprin et al. (2019) and Sengupta and Pedersen (2019) employing such procedures for the cleaning of hyperpartisan articles.

Feature Stemming: Moreover, stemming reduces each word to its root form by removing its suffix. In doing so, words which inherently carry the same meaning yet are written differently are reduced to their common root, thereby reducing the data complexity and removing any possibility of the words being understood, and treated, differently. Stemming is adapted to the field of hyperpartisan news classification by systems such as Cruz et al. (2019) and Palić et al. (2019).

Porter's Stemmer [Jones and Willett (1997); Porter et al. (1980)] and Snowball [Porter (2001)] are two well-known stemming algorithms. Simply removing the suffix from the word, Porter's Stemmer is considered as a more basic (nonetheless effective) stemmer. Snowball (also known as Porter2) is universally considered as a better, more aggressive stemmer to Porter's [Wiese et al. (2011)]. It does not simply remove the suffix of the word, but also considers the context of the word and the lexical rules of the language in which it is written. Moreover, multi-language support and faster execution times to its predecessor are features of Snowball.



Porter's Stemmer

console		consol
consoled		consol
consoles		consol
consolidate		consolid
consolidated		consolid
consolidating	=>	consolid
consoling		consol
consolingly		consolingli

Snowball Stemmer

console		consol
consoled		consol
consoles		consol
consolidate		consolid
consolidated		consolid
consolidating	=>	consolid
consoling		consol
consolingly		consol

Figure 2.3: A sample of few words reduced to their stem using the Porter's and Snowball Stemmers. Source: *The English (Porter2) stemming algorithm*[7].

Feature Lemmatisation: An effective alternative system to stemming is lemmatisation. Lemmatisation makes use of lexical databases such as WordNet [Miller (1995)] to remove inflection from a given word and reduce it to its lemma (dictionary form), typically achieving better results than stemming [Balakrishnan and Lloyd-Yemoh (2014)]. It is a more elaborate process than stemming since the Part-of-Speech (POS) of each word is taken into consideration.

This added complexity is however often preferred to stemming, with studies such as Joo and Hwang (2019) and Palić et al. (2019) effectively incorporating lemmatisation in order to reduce the complexity of terms making up news articles provided for hyperpartisan classification. In building the HyperPT system, we decided on employing lemmatisation due to its reliability and ready to use NLTK support for the WordNet lemmatiser[8].

2.2.2 | Feature Representation for Hyperpartisan News Articles

Having discussed data preprocessing adapted within HyperPT, we now examine some of the most prominent and widely used feature representation approaches for textual data in the field of NLP and more specifically the detection of hyperpartisan articles. In Section 2.2.2.1 we first examine traditional approaches, giving a concise overview of each. Approaches discussed are the BoW model, TF-IDF and POS-Tagging. This is followed by Section 2.2.2.2 where we discuss word embedding technologies for DL models, namely Word2Vec, GloVe, ELMo, and finally BERT.

2.2.2.1 | Traditional Feature Representation Approaches

Bag-of-Words (BoW): A very common approach for textual features representation is stripping a textual body of its structure, representing it as a list of unique words coupled with their frequency of appearance throughout the document. This is known as the Bag-of-Words (BoW) model and is a very popular feature representation method for textual data [Amason et al. (2019); Bestgen (2019); Potthast et al. (2018); Saleh et al. (2019)]. The BoW model is easy to understand, simple to implement and an effective baseline approach for the representation of textual features. Due to the simplistic nature of the BoW model, one does not get any insight into the importance of textual features besides the number of occurrences of each word contained within. Therefore unique, important words and common unimportant ones are treated with the same prominence - if not less due to the more frequent occurrences of the latter.

[7] *The English (Porter2) stemming algorithm* - `snowball.tartarus.org` [Last Accessed: 07-2020]

[8] *NLTK Stemming & Lemmatisation* - `www.nltk.org` [Last Accessed: 07-2020]

Term Frequency - Inverse Document Frequency (TF-IDF): TF-IDF [Jones (1972); Wu et al. (2008)] is a feature representation which reduces the importance value of a term the more common it is throughout all of the corpus, thereby highlighting words which are common to a document yet rare across the rest of the dataset as important, and common words all throughout as insignificant.

TF-IDF calculates the weight of each feature based on two calculations. First, the normalised term frequency (TF) inside of the document is computed. To do so we take the number of occurrences of a term t inside of a document d, and divide it by the total number of words (t') inside the same document. The TF measure [Salton and Buckley (1988)] is formally defined in Equation 2.1.

$$tf(t,d) = \frac{f_{\mathrm{t,d}}}{\sum\limits_{t'} f_{\mathrm{t',d}}} \tag{2.1}$$

The second measure is the inverse document frequency (IDF). It is defined by the log of; the total number of documents divided by the number of documents containing the term t. Thereby we give weight to the term depending on the amount of documents to contain it. If a word is very common among the set of documents, the weight given will be small, while if the contrary, the weight will be large (implying more importance).

Formally, the IDF calculation is given as shown in Equation 2.2, where we have the term t, the total number of documents D and the number of documents containing t, d_{t}.

$$idf(t) = log\frac{D}{d_{\mathrm{t}}} \tag{2.2}$$

Finally, TF and IDF are multiplied together to obtain the TF-IDF measure, as displayed in Equation 2.3.

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D) \tag{2.3}$$

TF-IDF is one of the most widely used traditional feature representation approaches for textual features. It is often preferred due to being a simple calculation with notably effective capabilities of distributing weight to salient words. Within the hyperpartisan news detection community, we find TF-IDF prominent as a reliable baseline system with which other experimental procedures are evaluated [Alabdulkarim and Alhindi (2019); Shaprin et al. (2019)].

Part-of-Speech (POS) Tagging: POS-Tagging [DeRose (1988); Petrov et al. (2011)] involves labelling each term depending on its morphological properties and context. Words can be labelled as verbs, nouns, pronouns, and so on according to their lexical nature. The use of Hidden Markov Models (HMMs) [Charniak (1997)] is an early effort at labelling POS-tags based on calculating the probability of likelihood that the upcoming word is of a certain part-of-speech. Moreover, rule-based POS-Tagging [Brill (1992, 1994)] systems such as Gupta et al. (2011) are well-known, finding their use not only in the English language, but also for more exotic languages such as Hindi.

Nowadays NLTK [Bird et al. (2009); Loper and Bird (2002)] offers an off-the-shelf POS-Tagger using the Penn Treebank [Marcus et al. (1993)] annotated corpus for English, and is hence widely used due to its flexibility and ease of use, with systems such as Nguyen et al. (2019) applying POS-Tagging within hyperpartisan news classification.

2.2.2.2 | Contextualised Word Embedding Feature Representation Approaches

Traditional feature representation systems struggle to properly identify a word's characteristics, be it its context in a document, its semantic or syntactic similarity with other words. Word Embeddings is an approach capable of representing text features in such a way that semantically and syntactically similar words are represented with similar feature vectors, while features not sharing such similarity are represented by correspondingly distant vectors. In this way, another level of detail is added to the corpus features which does not exist in traditional representation methods, with the expectation of improving classifier performance [Hettinger et al. (2018)].

Word embeddings are optimised along with the weights of a Neural Network during training. Dense vectors representing textual features are modified until a sufficient representation is achieved. Moreover, word embeddings are heavily used in DL due to their preservation of detail and natural compatibility with such classification algorithms. Throughout this section, we take a look at some of the most prominent Word Embedding technologies currently in use.

Word2Vec: The Word2Vec system is perhaps the most well-known word embeddings model at the time of writing - commonly employed as well for the classification of hyperpartisan news articles [Agerri (2019); Joo and Hwang (2019); Stevanoski and Gievska (2019); Zehe et al. (2019)]. Introduced by Mikolov et al. (2013a,b), it has gained significant popularity since. The Word2Vec model is pre-trained on the Google News dataset consisting of more than 100 billion words [Mikolov et al. (2013a)], making it a notably knowledgeable model capable of catering to virtually any form of document written in the English language.

In practice, one finds two adaptations of the Word2Vec model, the Continuous Bag-of-Words (CBOW) model (featured in Figure 2.4) and the Skipgram model. Albeit being different, at their core they both consist of a two-layer neural network with a Softmax activation function, trained on rebuilding the linguistic contexts of given features [Mikolov et al. (2013a,b)]. The two systems function inversely to one another; the CBOW model predicts a target word from the surrounding context words it is given, while Skipgram predicts the context surrounding a given word. Moreover through the use of hierarchical softmax and negative sampling, computation time is optimised. This is since in hierarchical softmax, all words are represented as a binary tree with probabilities of the representation of words at the leave nodes calculated along the tree paths. As for negative sampling, only a sample of contextual few words is updated (negatively sampled) at each iteration [Rong (2014)].

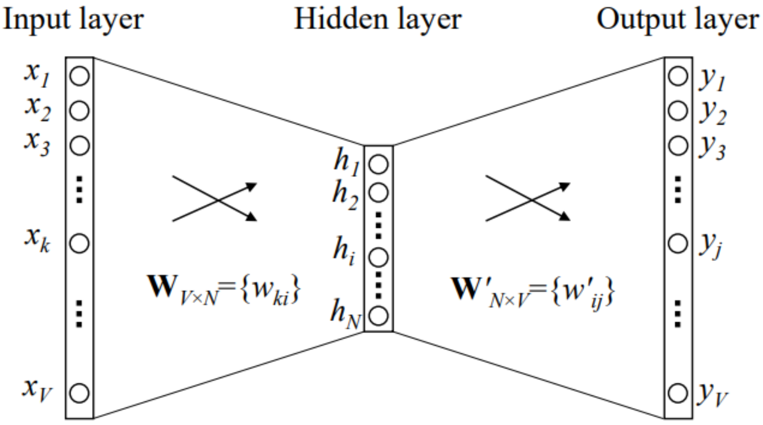


Figure 2.4: A high level visualisation of the Word2Vec Continuous Bag-of-Words (CBOW) Architecture. Source: Rong (2014).

The Word2vec Skipgram model utilises the local context around a target word to obtain the corresponding vector. This makes it ideal for analogical tasks such as *"king is to queen as man is to woman*, which results in the vector *king* - *queen* = *man* - *woman* [Mikolov et al. (2013a)]. It performs poorly however when taking into consideration global statistics on the corpus, since Word2Vec trains using local context windows and not on the global co-occurrence counts of the features.

Global Vectors (GloVe): In order to address this issue, Pennington et al. (2014) introduce GloVe. GloVe is a global log-bilinear regression model which produces word embeddings by combining the advantages of global matrix factorisation (such as the

Latent Semantic Analysis [LSA] model by Deerwester et al. (1990)) and local context window methods (as discussed, the Word2Vec Skipgram model). The model trains on the non-zero elements in a word-to-word co-occurrence matrix, reducing the complexity in dealing with sparse vectors and in doing so speeding up the training process. Hence, it is able to evaluate the likelihood (as probability) of two words appearing together, thereby determining whether a feature i is only common with another feature j, or common all throughout. From these values the relationships formed between the corpus words could be deciphered.

Embeddings from Language Models (ELMo): Introduced by Peters et al. (2018), ELMo is a deep, contextualised word representation system capable of modelling both 1) complex characteristics of word use, and also 2) how these uses vary across linguistic contexts, hence providing context-based feedback for a given corpus.

Similar to the Word2Vec CBOW model [Mikolov et al. (2013a,b)], ELMo predicts the upcoming target token, given a context of tokens. It does so using a Bidirectional Long Short-Term Memory (biLSTM) neural network, with one LSTM scanning the given sentences from left to right, and the other from right to left. The LSTMs hence compensate for each other's decreasing attention the further from the starting point they are. The biLSTM is moreover trained with a coupled Language Model (LM) on large volumes of textual data.

In this way, ELMo operates differently to other, less elaborate word embedding technologies such as Word2Vec and GloVe. Such systems generate a single, context-independent representation for each target word, while ELMo representations consist of a function of all the internal layers inside the biLSTM. As the authors themselves claim, the combination of the LSTM's internal states creates rich word representations, where higher-level LSTM states capture context-dependent word meanings, and lower-level LSTM states capture the model's syntactic aspects.

ELMo is pretrained over 10 epochs on the *1B Word Benchmark* [Chelba et al. (2014)], a state-of-the-art benchmark corpus for statistical language modelling. Consisting of two biLSTM layers with 4096 units and 512 dimension projections, as well as a linear projection layer, the ELMo architecture produces three outputs per feature, corresponding to each of the three layers. The outputs can be utilised individually or aggregately, depending on the nature of the application - with systems like Jiang et al. (2019) averaging the three representation vectors in order to combine the benefits of all the outputs.

Bidirectional Encoder Representations from Transformers (BERT): An emerging competitor to the superior performance promised by contextualised word embeddings are transformer-based solutions. BERT, a new language representation model developed Devlin et al. (2019), is one such system.

The Transformer is a neural network architecture introduced by Vaswani et al. (2017) which reportedly offers added benefits and improvements on RNN and LSTM networks. Transformers find their beginnings as encoder-decoders. Different to the typical encoder-decoder architecture however, transformers focus entirely on the attention given to the corpus, replacing recurrent layers with attention mechanisms - more formally known as multi-headed self-attention [Vaswani et al. (2017)].

Since the goal of the algorithm is to generate a language model for the given corpus, BERT uses solely the encoding part of the transformer architecture. A level of masking, in which 15% of the words passed are hidden, is integrated within the encoding layers in order to introduce more stochasticity inside of the network and encourage learning efforts (while also decreasing the possibility of internal overfitting).

BERT offers two systems (with smaller systems also made available at the time of writing) [Turc et al. (2019)]; BERT-Base and BERT-Large. BERT-Base consists of 12 transformer blocks and 768 hidden nodes with 12 attention heads, while BERT-Large is made up of 24 transformer blocks and 1024 hidden nodes, with 16 attention heads. Both variations are pretrained with the intention of generating bidirectional representations from unlabelled text. Through the use of attention mechanisms, training is conditioned simultaneously on all textual contexts inside of the corpus, making the model capable of fine-tuning with only one additional output layer [Devlin et al. (2019)].

2.3 | Classification Approaches for Hyperpartisan News Articles

Throughout this section we examine classification algorithms prominent in the field of text-based NLP. We discuss traditional ML approaches such as Logistic Regression (LR), Random Forest (RF) and Support Vector Machine (SVM) in Section 2.3.1, exploring the reasoning and logic behind each method, before repeating the process on the more elaborate DL methods, namely Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) - featured in Section 2.3.2.

2.3.1 | Traditional Approaches

Traditional methods employed within the research area of hyperpartisan detection range from well-known and widely used techniques to other less popular and niche approaches.

Popular methods include the LR - widely used both within the domain of ML and the area of hyperpartisan news articles. Moreover, the SVM is a popular more advanced alternative. Although typically being less common, the RF classifier is also found adapted to such problems, offering a completely different approach to the previous two. Throughout this section we examine these three traditional ML classification approaches, discussing their differences, similarities and expected performance.

Logistic Regression (LR): LR is a simple method, yet an effective classifier to most basic and common binary (0 or 1) classification problems. Underneath, the Maximum Likelihood Estimation (MLE) is computed in order to find the model's best fit on the give data, until a convergence criterion is reached.

$$Sigmoid(x) = \frac{1}{1+e^{-x}} \tag{2.4}$$

The LR model separates two classes by forming an S-shape curve transitioning from 0 up to 1 - formally known as a Sigmoid function (Equation 2.4). The value generated by LR can be considered as the probability of the input being in one of the two classes. If the output nears 0, the probability of being in the hypothetical class A is very small - implying that the input sample belongs to class B. The opposite holds true if the output nears 1, with the probability being that the input sample fits with class A and not class B.

Support Vector Machine (SVM): The LR classifier is simple and quick, yet struggles to fit more complex multi-dimensional problems. The SVM [Cortes and Vapnik (1995); Vapnik (1998)] introduces the hyperplane concept for more elaborate non-linear applications. Put simply, the goal of the SVM is to find the optimal hyperplane in a multi-dimensional space separating different classes of data points (input samples). In doing so, a classification system is created which given a new data sample, can determine with which class it is most similar depending on its position in the hyperspace in relation to the SVM hyperplane.

As its name implies, the SVM makes use of what are known as support vectors to determine the best fit to a classification problem. Support vectors can be understood as the data points forming the smallest margin with the SVM's separator line. The support vectors are the only points taken into consideration and they support the SVM hyperplane itself (as visualised in Figure 2.5). During the training phase, the hyperplane is optimised in order to maximise the margin as much as possible. A degree of tolerance is often specified, allowing for a number of data points to breach the hyperplane margins.

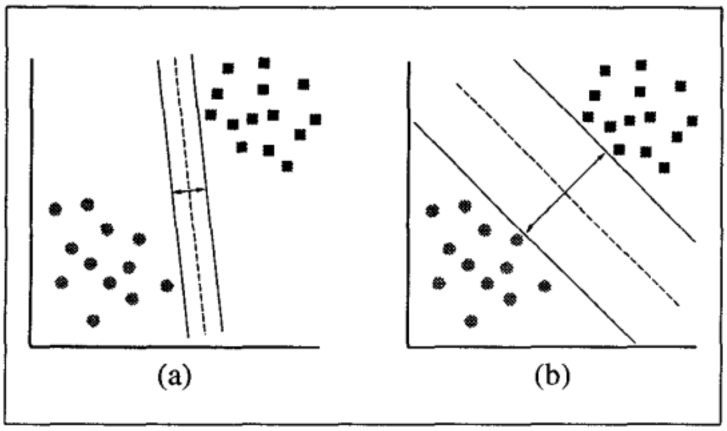


Figure 2.5: An example of two hyperplanes, with the second being more spread than the first. The second hyperplane is expected to have better generalisation in its classification due to the larger area covered. Source: Osuna et al. (1997).

The optimisation of the hyperplane is performed by transforming the problem using what is known as a kernel function [Amari and Wu (1999); Fletcher (2009)]. In other words, the kernel maps the given data points to the SVM's hyperspace. One finds a number of possible kernels, with more complex ones being capable of mapping data points to higher dimensions.

The most basic kernel would be Linear, which simulates the same behaviour of the Logistic Regression (LR) classifier. The Sigmoid kernel on the other hand finds its bases in the same Sigmoid function at the heart of the Logistic Regression (Equation 2.4). One finds the Polynomial kernel, which contrary to the Linear kernel, supports polynomial functions. Moreover, kernels of exponential (e^x) nature such as the Radial Basis Function (RBF) [Amari and Wu (1999)], are typically capable of fitting any classification problem as much as the previous three kernels, along with more elaborate and complex cases. Another general purpose exponential function would be the Gaussian Kernel. Given such a selection of kernel functions, one must perform experimentations with the data at hand and determine objectively which kernel tends to fit the problem best.

Random Forest (RF): The RF classifier, introduced by Breiman (2001), is based on the principle that the correct opinion of a large number of uncorrelated individuals outperforms the mistaken opinion of some individuals. It is made up of a number of Decision Trees, each of them classifying random samples picked out from the dataset, and basing their predictions on different features. Finally the global opinion on a data sample is evaluated, and the sample is classified according to the most common opinion among the forest of Decision Trees.

Known also as CART (Classification And Regression Trees), the decision tree model is inherently a binary tree. During its inception, the input is split until a satisfactory tree model is assembled. To measure the quality of each split a specific function is used which computes the impurity at each node. Typically, one of the two following functions is used; the Gini Index impurity or the Entropy information gain (Equation 2.5 and Equation 2.6 respectively, where $p(c_{\text{i}})$ is the probability of class c_{i} in a given node). Albeit the two being quite similar, Gini is typically preferred due to it being less computationally expensive [Raileanu and Stoffel (2004)].

$$Gini = 1 - \sum_{i=1}^{n} p^{2}(c_{\text{i}}) \tag{2.5}$$

$$Entropy = \sum_{i=1}^{n} -p(c_{\text{i}}) \log_{2}(p(c_{\text{i}})) \tag{2.6}$$

Considered as a high-variance model [Dietterich and Kong (1995)], decision trees are more often than not used in an ensemble method called Bootstrap Aggregation [Breiman (1996)] - or Bagging. In such an organisation, multiple decision trees are initialised, with all the corresponding outputs being averaged such that the most common prediction within the group of decision trees is agreed upon as the final output. In this way, more stability is introduced to the final prediction which would have otherwise not been possible with an individual decision tree.

Variable splitting at each tree is performed in a greedy manner, with individual trees ending up splitting at the same variable locations - resulting in very similar outcomes. Random Forest is hence an improvement on the classical Bagging approach, which limits each sub-group of trees to a set of variables among which to perform their splits, thereby limiting the trees to their own set of variables. In doing so, variety is forced between the tree population, increasing the model's generalisation capabilities while maintaining the same performance [Breiman (2001)]. Moreover, the RF classifier tends to be stable in its performance, capable of resisting overfitting on the training data due to the Bagging approach employed within.

2.3.2 | Deep Learning Approaches

Having discussed some of the most prominent traditional ML approaches, we now discuss Deep Learning (DL) classification algorithms for the classification of text-based NLP tasks, in particular the classification of hyperpartisan news articles. In summary, the DL scene within hyperpartisan news detection is dominated by CNN and RNN architectures. Ensemble models utilising both technologies are also quite common, with CNN-Bagging, biRNNs and biLSTMs often evaluated against the performance of one another. We hence discuss a concise background on these technologies and examine the underlying functionality which makes them so effective within the corresponding research area.

Convolutional Neural Network (CNN): The CNN classifier is perhaps more well known for its distinctive ability to adapt to Artificial Vision and image applications. However, as shown in Figure 2.6, textual sequences such as sentences and documents can be passed to a CNN in much the same way an image can - as long as the feature representation is compatible with the network. The CNN consists of three main parts [O'Shea and Nash (2015)]; 1) the convolutional layers, 2) the pooling layers and finally 3) the fully connected layers.

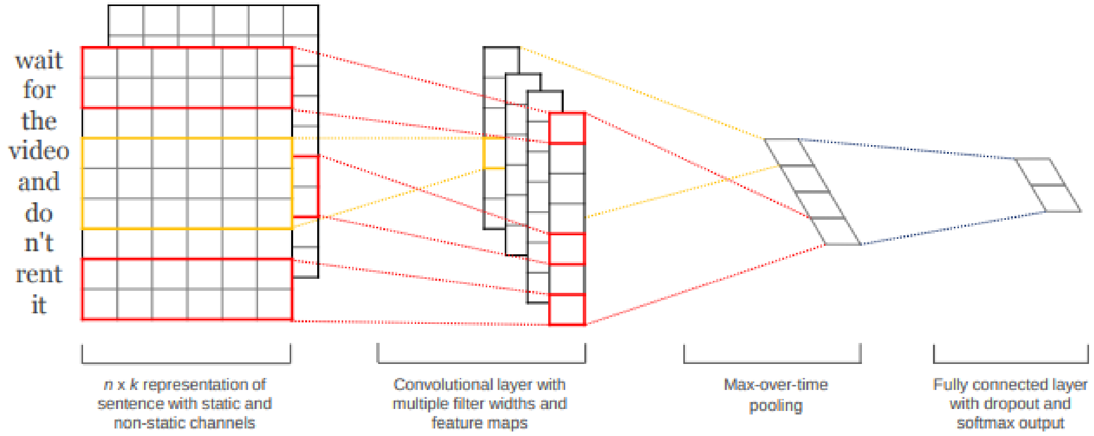

wait
for
the
video
and
do
n't
rent
it
n x *k* representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 2.6: A high level visualisation of a simple sentence being processed by a Convolutional Neural Network (CNN). Source: Kim (2014).

One considers the convolutional layers as the most distinctive features of the CNN model. As the name itself implies, convolution functions are applied on the input data. Through the use of convolution filters, a feature map of the input is generated - highlighting the most important properties of the given image or text sequence. For 2-dimensional data inputs such as images, convolution filters are applied in the form of a window being passed over each section of the input, while in the case of 1-dimensional textual data, the same process is performed, yet with a 1-dimensional vector of weights passing through the sequence of text. In practice one typically finds CNN applying several filters on the input, reducing the data dimensionality and extracting the most important features. The summarised feature vectors are then passed on to the pooling layers.

The role of the pooling layers is to compress and generalise the features extracted by the convolutional layers - preventing the network from overfitting on the training data - which is particularly a problem when it comes to the location of the features inside of the input image or text sequence. Pooling makes sure that the data is generalised enough such that the classification layers could still recognise distinctive features located at different parts of the input.

Pooling is generally applied after each convolutional layer - with an activation function such as ReLU (Equation 2.7) being applied in between for non-linearity. One of two pooling functions is typically used; average or max pooling. In case of average pooling, the average of the values within the local area is calculated, while for the max pooling, the maximum value of the lot is selected.

$$ReLU(x) = max(0, x) \tag{2.7}$$

Finally, the processed data is passed to the fully-connected layers, which perform the actual classification on the pooled feature maps generated from the original input. A final activation function such as Sigmoid (Equation 2.4) is utilised, outputting a value representing the classification between 0 and 1. It is interesting to note that during backpropagation, the weights inside of the convolutional layers are updated along with the rest of the network, with the aim of enhancing the convolution filters.

Recurrent Neural Network (RNN): Different to the CNN architecture, RNNs [Chen (2016)] are designed around the concept of memory and remembrance. The model is capable of remembering the context a data sample is in, and train itself on such a dependency. It is specifically designed to do so for data sequences such as a sample of text. In practical terms, the nodes within the neural network have the added possibility of looping their signal back to themselves or neighbouring nodes. Moreover, the output of an RNN classification may be passed back again along with new input. This added complexity allows the network to communicate the context of an input i_1 back to itself.

RNNs are used in multiple areas of research, yet often find their place in the field of NLP due to their capability of maintaining context within a sequence of input. Due to its complex architecture, updating the network weights is not as straight-forward as it is for other neural networks, with classical backpropagation not being feasible.

Instead, Backpropagation through Time (BPTT) is used; unrolling the network nodes such that the recurrent links within the network are created as copies of individual node instances. The RNN is hence represented as a classical feed-forward neural network, and the weights are then updated accordingly.

The classical RNN architecture suffers from a major drawback. In updating the weights of deep unrolled neural networks, the gradients upon which the weights are calculated tend to become unstable, approaching either very low (vanishing gradient) or very large (exploding gradient) ranges. This in turn makes the network unstable and unreliable, while also extending the training time by a large margin.

Long Short-Term Memory (LSTM) Network: LSTM [Hochreiter and Schmidhuber (1997)] networks are an improvement on the RNN architecture - specifically designed to address the problem of vanishing and exploding gradients. LSTMs substitute the otherwise neural nodes with smarter memory blocks - also known as cells. Each cell contains three gates; input, output and forget gate [Gers et al. (1999)]. The input gate decides on which information to update the memory state with, while the output gate maintains the output - conditioned on the input and existing unit memory. The forget gate handles discarded information. Each of these three gates has its own weights which are themselves trained with the rest of the model.

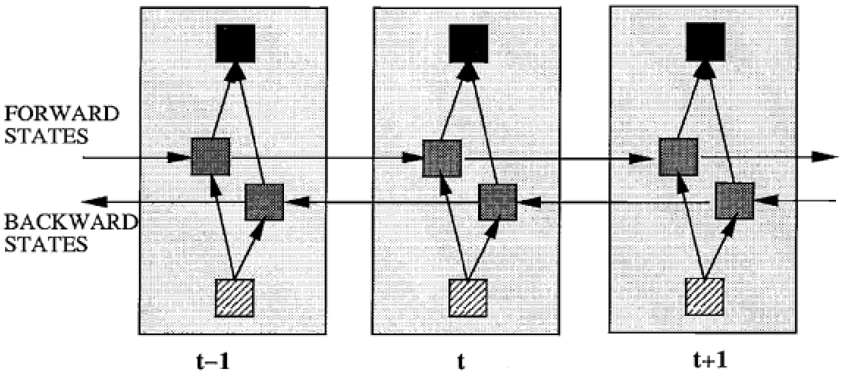


Figure 2.7: A simplified unfolded visual representation of the basic architecture of a Bidirectional Recurrent Neural Network (biRNN). Source: Schuster and Paliwal (1997).

Both RNNs and LSTMs process sequences of data sequentially, with concurrent processing not being possible. Due to the tendency of the network's attention to degrade with the length of the sequence, RNNs and LSTMs are typically used in a bidirectional ensemble, in which one network parses the sequence from left to right, and the other from right to left, thereby dedicating the same amount of attention for both ends of the given sequence [Schuster and Paliwal (1997)]. These models are known as bidirectional RNNs (biRNN) and LSTMs (BiLSTM). A high level overview of a biRNN model architecture is displayed in Figure 2.7.

2.4 | Sentiment Analysis in Hyperpartisan News Articles

Sentiment analysis is an important and well-researched domain in the field of NLP. It addresses the detection and extraction of opinions, sentiments and emotions towards a subject, be it an individual, an organisation, or any other entity. The sentiment within a sequence of text is often established through sentiment-bearing terms, their polarity, and the context in which they are used [Yadav (2015)].

Due to the sensationalised and dramatic elements that hyperpartisan articles try to pass on to the user, we investigate whether sentiment plays a prominent part in the detection on such news. Hereby, we discuss a concise background on the analysis of sentiment before considering it within the HyperPT system.

2.4.1 | Sentiment Analysis - Overview

One finds three families of sentiment classification systems; Lexicon-based approaches, ML approaches and Hybrid approaches [Yadav (2015)]. Lexicon-based approaches consist of a language lexicon (collection of known sentiment terms), usually including the synonyms and antonyms of each term. A lexicon can be pre-assembled from other corpora and *shipped* as is (dictionary-based approach), or it could be expanded and built on a specific corpus in real-time (corpus-based approach). The latter is implemented by having an already-established list of sentiment-bearing words, upon which statistical and semantic methods are then applied to expand the lexicon's vocabulary.

ML techniques are divided between supervised and unsupervised approaches. Supervised approaches are trained on labelled training corpora, where evaluation criteria correct the algorithm's performance. We find studies such as Pang et al. (2002), where Naive Bayes (NB), Maximum Entropy Classification (MaxEnt), and Support Vector Machines (SVMs) are evaluated for the classification of sentiment - with SVMs being the best performer. Moreover, the SVM is compared with ANN systems by studies such as Moraes et al. (2013), where the authors compare the performance of the two classifiers on movie and product reviews. The SVM comes at a close second with a highest accuracy of 85.2% to the ANN's 86.5% (respectively achieved with 1000 and 3000 features).

DL systems are also prominent in the field of sentiment analysis [Zhang et al. (2018)], with encoders [Glorot et al. (2011); Yin et al. (2017)] and LSTMs [Xu et al. (2016); Zhou et al. (2016)] being two such well-known classification approaches.

Unsupervised approaches are preferred in cases where it is difficult to obtain quantities of high quality labelled data upon which to train. Unsupervised sentiment classification tends to be heavily based on already known words, with methods such as Turney (2002) using known opinionated words and phrases. Paltoglou and Thelwall (2012) propose an unsupervised, lexicon-based sentiment analysis system which is less dependant than ML-based techniques on the domain in which it is used. The proposed system is tested on Twitter, MySpace and Digg text snippets, estimating the levels of emotional intensity found within the texts.

Hybrid approaches combine both ML and lexicon based techniques to create a system in which ML models are trained on gold-standard lexicons upon which given text is then classified. Due to the pivotal performance offered by language lexicons, such systems are well-known and effective [Yadav (2015)].

Sentiment analysis is typically applied at three levels of granularity; 1) Document-level, Sentence-level, and Aspect-level. At document level, a single sentiment is given for a document of text, while for sentence-level analysis, the same is repeated for each sentence. Aspect-level sentiment analysis however is more complex - assuming that a document of text contains various entities/objects, having their own aspects. Hence for successful sentiment classification, these objects and their contexts must first be taken into consideration [Yadav (2015)].

2.4.2 | VADER for Sentiment Analysis

VADER (Valence Aware Dictionary for sEntiment Reasoning) is a lexicon-based sentiment analysis system introduced by Hutto and Gilbert (2014). The researchers make use of qualitative and quantitative techniques along with grammatical and syntactical conventions for the expression of sentiment, in order to create a gold-standard sentiment lexicon. To our knowledge, VADER is one of the most prominent off-the-shelf sentiment analysis systems currently available.

The authors' aims in designing such a system were to build a computational sentiment analysis machine that is generalisable and reliable on different domains and styles of writing, while also performing well on social media styles of text. Moreover, the lexicon requires no training data since it is assembled from a valence (intensity) based gold standard sentiment dictionary created using human expert analysis. Compared to alternative methods such as ML-based techniques, VADER is considered to have notably lower execution times [Hutto and Gilbert (2014)].

As discussed above, VADER is based on expertly selected sentiment lexicons. Having been inspired by such word dictionaries (LIWC[9], ANEW[10] and GI[11]), multiple expressions from microblogs and social media texts were incorporated - amounting to over 9000 lexical features. Wisdom of the crowds [Surowiecki (2004)] was then used to acquire estimates for the sentiment valence of each of these features. Finally, these were reviewed by ten independent human raters. Moreover, 400 positive and 400 negative expertly-selected tweets were used to generate five generalisable heuristics conveying the intensity of the sentiment.

VADER is evaluated against other well-known ML and lexicon based systems, namely LIWC, ANEW, WSD, SCN, GI and Hu-Liu04 - with ML methods being Naive Bayes (NB), SVM, and Maximum Entropy (ME). It performs exceptionally well, surpassing any system (including human analysis) on social media text, while maintaining second position (exceeded by human analysis) for other datasets such as Amazon Product Reviews and Movie Reviews. Being offered out-of-the-box by NLTK, the VADER sentiment lexicon is widely used for the analysis of sentiment, including the field of hyperpartisan news article detection [Anthonio and Kloppenburg (2019); Joo and Hwang (2019)].

In practice, a string of text can be instantly passed to the VADER object instance, which would return the corresponding sentiment values for four sentiment measures; *negative*, *neutral*, *positive* and *compound*. The compound score can be considered as a more context-sensitive sentiment measure, taking into consideration the valence of the word within the context in which it is used. It is given as a number between -1 to 1, with the former being extreme negative and the latter being extreme positive.

2.5 | Explainable Artificial Intelligence for Hyperpartisan News Classification

Explainable Artificial Intelligence (XAI) [Arras et al. (2016); Goebel et al. (2018); Samek et al. (2017)] is a new and emerging branch of the A.I. research and development community focused on the interpretation and explanation of the logic behind an A.I. system's behaviour. In our study, we aim at incorporating explainable A.I. for the interpre-

[9] ***LIWC*** - `liwc.wpengine.com` [Last Accessed : 07-2020]

[10] ***Affective Norms for English Words (ANEW)*** - `csea.phhp.ufl.edu` [Last Accessed : 07-2020]

[11] ***General Inquirer (GI)*** - `www.wjh.harvard.edu` [Last Accessed : 07-2020]

tation of the model's decision-making, in doing understanding the model's reasoning while also examining the features influencing it's decisions.

Two recent techniques referred to for the explanation of classifiers' predictions are Sensitivity Analysis (SA) and Layerwise Relevance Propagation (LRP) [Samek et al. (2017)]. Albeit being inherently different, both algorithms produce a similar set of results: an individual score for each input feature passed to the classification system - representing the amount of relative influence of each feature on the class prediction. Through such scores, one could observe which features affect the classification, and how a classification label is derived from a sequence of inputted features. Both algorithms are typically applied on Neural Network based architectures, but are adapted to other traditional ML classifiers as well [Arras et al. (2017)].

2.5.1 | Sensitivity Analysis

Sensitivity Analysis (SA) [Baehrens et al. (2010); Samek et al. (2017)] attempts at explaining a classifier's prediction based on the model's locally evaluated gradients (partial derivatives). Features having the most sensitive output are considered as the most relevant. Hence, in formal terminology, the relevance R_i of each input feature i is given as shown in Equation 2.8, where $f(x)$ is the classification function.

$$R_\mathrm{i} = \| \frac{\vartheta}{\vartheta x_\mathrm{i}} f(x) \| \tag{2.8}$$

As published literature points out [Montavon et al. (2018); Samek et al. (2017)], SA has the inherent limitation of not explaining the function value $f(x)$ itself, but a variation of it. It highlights which features need to be changed the most (from the model's perspective) for an input to be closer to the predicted class. Due to this fact and other setbacks (discussed below) inherent to the SA interpretability procedure, it is often considered as a primitive baseline model, having been proven inferior to alternative techniques [Samek et al. (2017)].

2.5.2 | Layerwise Relevance Propagation

One such technique showing superiority over SA is Layerwise Revelance Propagation (LRP) [Bach et al. (2015); Samek et al. (2017)]. LRP attempts at interpreting a classifier's decisions through decomposition. The classifier's prediction is redistributed backwards using local redistribution rules until eventually a relevance score is assigned to each input feature. At every step of the redistribution the total amount of relevance is preserved. LRP explains the classifier's prediction corresponding to the state of maximum uncertainty - in other words identifying the crucial features affecting a classification.

Having been originally developed for pixel-level model evaluation in computer vision tasks [Bach et al. (2015)], LRP bases its computations on the hypothesis that there exists a relevance score $R_d^{(l+1)}$ for each dimension $z_d^{(l+1)}$ of the vector z at layer $l+1$. The algorithm hence attempts to maintain the relevance conservation $R_d^{(l)}$ for each dimension $z_d^{(l)}$ at the ensuing layer l as shown in Equation 2.9 - which states that at any step of the redistribution procedure, the total amount of relevance is maintained [Bach et al. (2015); Samek et al. (2017)].

$$f(x) = \sum_{d \in l+1} R_d^{(l+1)} = \sum_{d \in l} R_d^{(l)} = ... = \sum_{d} R_d^{(1)} \tag{2.9}$$

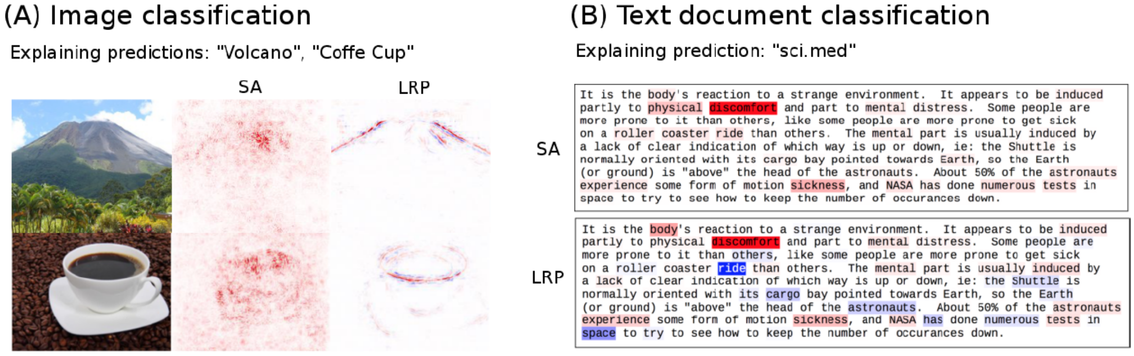
LRP redistributes the relevance from layer $l+1$ to layer l of the model as given in Equation 2.10, where x_{j} are the neuron activations at layer l (assuming the classifier is a neural network), R_{k} represents the relevance scores corresponding to the neurons at later $l+1$, and w_{jk} being the weight between neuron j and neuron k. Note also that a small stabilisation term ϵ is included to avoid the possibility of division by zero. Through this calculation, relevance is distributed according to the neuron activation x_j (with a larger share of relevance given to the more activated neurons), and the strength of the connection between neurons j and k, with more relevance given to more pivotal weight connections.

$$R_{\mathrm{j}} = \sum_{k} \frac{x_j w_{jk}}{\sum_{j} x_j w_{jk} + \epsilon} \tag{2.10}$$

LRP is also capable of analysing traditional ML models such as SVMs [Arras et al. (2017); Bach et al. (2015)]. Assuming that w_c and b_c are respectively class-specific weights and biases, and D is the number of non-zero vectors representing BoW documents; the relevance decomposition R_j for an input feature x_j is computed as given in Equation 2.11 [Arras et al. (2017)].

$$R_j = (w_c)_j \cdot x_j + \frac{b_c}{D} \tag{2.11}$$

Contrasting to SA, LRP decomposes the actual function value $f(x)$. Further to this, LRP has the capability of determining whether each feature supports or opposes a particular classification, while SA limits itself to solely determining how relevant a feature is to a particular class. Having been evaluated by studies such as Arras et al. (2017) and Samek et al. (2017), a clear discrepancy in interpretability performance is indeed observed. In Figure 2.8, Samek et al. (2017) produce heatmaps of SA and LRP interpretability on image and text classifications. Two observations stand out; the clearer plotting of the object's edges in case of the image classification heatmap, and the addition of features opposing the classification (highlighted in blue) in the text heatmap.



It is the body's reaction to a strange environment. It appears to be induced partly to physical discomfort and part to mental distress. Some people are more prone to it than others, like some people are more prone to get sick on a roller coaster ride than others. The mental part is usually induced by a lack of clear indication of which way is up or down, ie: the Shuttle is normally oriented with its cargo bay pointed towards Earth, so the Earth (or ground) is "above" the head of the astronauts. About 50% of the astronauts experience some form of motion sickness, and NASA has done numerous tests in space to try to see how to keep the number of occurances down.

It is the body's reaction to a strange environment. It appears to be induced partly to physical discomfort and part to mental distress. Some people are more prone to it than others, like some people are more prone to get sick on a roller coaster ride than others. The mental part is usually induced by a lack of clear indication of which way is up or down, ie: the Shuttle is normally oriented with its cargo bay pointed towards Earth, so the Earth (or ground) is "above" the head of the astronauts. About 50% of the astronauts experience some form of motion sickness, and NASA has done numerous tests in space to try to see how to keep the number of occurances down.

Figure 2.8: Image and Text classification interpretability - comparison of Sensitivity Analysis (SA) and Layerwise Relevance Propagation (LRP). Source: Samek et al. (2017).

Several ready to use libraries implementing SA and LRP are available. The LRP Toolbox [Lapuschkin et al. (2016)] offers LRP analysis adapted to Caffe networks and bespoke implementations of neural network models. Moreover, iNNvestigate [Alber et al. (2019)] is a library offering a range of analysis methods, including gradient-based solutions like SA and variations of LRP. In our system, HyperPT, we make use of iNNvestigate to implement the SA and LRP interpretability methods on our CNN classifier, however since at the time of writing we failed to find a reliable off-the-shelf library implementing the same on the SVM classifier, we developed the LRP algorithm on this algorithm from first principles, using the work of Arras et al. (2017) as guidance.

2.6 | Evaluation Criteria

Due to the nature of the task at hand and the SemEval Hyperpartisan News Dataset (Section 2.1), we evaluate the HyperPT system using the classification accuracy and F1 score. This decision is supported by two reasons;

1. Given that it is a classification problem, the ultimate performance review would be by examining the classifier's accuracy and balance in its predictions.

2. Other, similar systems evaluate along the same evaluation metrics, and hence in this way, we can compare our system performance with theirs.

The accuracy score is given as a decimal value between 0 and 1, thereby 0 implying 0% and 1 implying 100%. The F1 score (or F measure) is calculated on the Precision (Equation 2.12) and the Recall (Equation 2.13) scores of the classifier predictions as shown in Equation 2.15.

$$Precision = \frac{TruePositives}{TotalPredictedPositives} \tag{2.12}$$

$$Recall(Sensitivity) = \frac{TruePositives}{TotalActualPositives} \tag{2.13}$$

$$Specificity = \frac{TrueNegatives}{TotalActualNegatives} \tag{2.14}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.15}$$

The accuracy measure provides us with the ratio of correctly classified articles to the total number of articles considered within the classification. Albeit being a strong overall performance overview, accuracy fails in indicating potential biases in a classification system, where one class could be more susceptible to wrong classification than the other.

Conversely, the precision takes into consideration the number of true positives (correctly classified hyperpartisan articles) with respect to the total hyperpartisan predicted articles (including falsely predicted as so). Moreover, recall (also known as sensitivity) is the number of true positives divided by the total number of actually positive (hyperpartisan) news articles available within the classification. Hence a low precision score suggests a high number of false positive classifications (wrongly classified as hyperpartisan), while a low recall score implies that a low number of true positive (hyperpartisan) articles are correctly classified as so, with the rest being falsely classified as negative (neutral).

Subsequently, the F1 score provides us with the harmonic mean of the precision and the recall. In practice, the F1 score is a value between 0 and 1 - with 0 being the poorest and 1 being the best, performance-wise. This coupled with the accuracy gives us a strong indication as to 1) the overall performance of the system and 2) How balanced and reliable the system is proportional to the data samples utilised within the classification.

To keep the HyperPT study to a manageable scale, given the limitations imposed on the research, we maintain the accuracy and the F1 score as the main evaluation criteria. We do not rule however, as future work, the consideration of other evaluation criteria in order to provide a more granular indication of the classification performance. The recall/sensitivity and the specificity (Equation 2.14), for instance, could be used to respectively determine the ratio of correctly classified hyperpartisan and neutral articles with respect to the total number of available hyperpartisan and neutral articles. In such a case, due to the harmful effects of widespread hyperpartisan news articles, one may want to focus on the sensitivity in order to verify that all possible hyperpartisan news articles are captured.

Separate to the classification system, in employing model interpretability algorithms such as Sensitivity Analysis (SA) and Layerwise Relevance Propagation (LRP), we evaluate the system by adapting a method introduced by similar studies [Arras et al. (2017); Samek et al. (2017)] in which the classification accuracy is monitored with the removal of each salient feature - from the most salient to the least. In this way, the rate of degradation in accuracy performance is directly related to the quality of the feature selections performed by the interpretability techniques.

2.7 | Related Work

Having examined the background behind techniques utilised within the HyperPT study, we now examine published literature employing the same or similar approaches. We commence in Section 2.7.1 by discussing the related area of fake news detection, exploring the work conducted by other researchers in tackling the spread of such malicious information.

This is followed by similar work on the specific classification of hyperpartisan news. Addressed in Section 2.7.2, we discuss traditional and deep learning approaches for the detection of hyperpartisan news. Moreover, we tackle the role of sentiment within this research and the addition of explainable A.I. - two prominent components investigated within this study.

2.7.1 | Detection of Fake News

Due to the growing need to control the spread of fake news, one finds an interesting area of study behind the development of intelligent techniques designed for its automatic detection. Throughout this section we examine published literature to discover the spectrum of systems tackling the neighbouring problem of fake news.

Potthast et al. (2018) classify the detection techniques of both fake and hyperpartisan news into three groups: i) Knowledge-based, ii) Context-based and iii) Style-based.

Knowledge-based systems compare individual pieces of knowledge claims inside of a given text with knowledge gathered from online systems such as webpages [Etzioni et al. (2008); Yates et al. (2007)] - computing the differences. Other, similar systems build statistical models, also based on knowledge gathered online [Magdy and Wanas (2010)]. Such systems depend heavily on the assumption that webpage sources from which knowledge is gathered are indeed reliable and accurate. Other approaches like Wu et al. (2014) make use of more reliable knowledge-bases to try and verify a given claim. By treating claims as queries with corresponding parameters, they can be checked not only for correctness and authenticity, but also for the quality of the claim.

Alternatively, context-based systems study how information is spread on social media, to then engineer ways for slowing down or stopping this spread [Agrawal et al. (2011); Nguyen et al. (2015)]. Tambuscio et al. (2015) conduct observations relating to how fake news articles are shared. It is claimed that increasing the threshold required to share a fake news article, such as the amount of convincing individuals needed, should be enough for the news article to die off before going viral. The researchers implement a stochastic epidemic model in order to describe the spread of a hoax piece of news on social media, proposing a correlation between the amount of fact-checkers needed and the complete removal of the hoax from online circulation.

Style-based fake news detection sets aside the need to verify the truth behind claims, focusing instead on the style of writing. Style-based approaches can be divided into two sub-groups; detection of deception within text and categorisation of text. Detection of deception builds on the Undeutsch hypothesis [Undeutsch (1967)] - stating that memories of real, experienced events differ from those of imagined events.

Studies such as Kwon et al. (2013) expand on this observation by studying the patterns of rumour spreading on the Twitter[12] social media. The researchers explore three sets of features; temporal, structural, and linguistic. A pattern of repeating spikes in activity for tweets with rumours is noted to not be present in non-rumour tweets. Moreover, the structure of the follower-followee connection graph of Twitter users is taken into consideration. Finally, the LIWC dictionary-based sentiment analysis tool (addressed also in Section 2.4) is used to detect and categorise the sentiment features within the tweets.

Rubin et al. (2015) address deceptive news by being the first system to apply deception detection techniques to news articles through the analysis of rhetorical structures, discourse constituent parts and corresponding coherence relations. A vector space model is used to cluster news by similarity of discourse, with an accuracy of 0.63. A predictive

[12] ***Twitter*** - `www.twitter.com` [Last Accessed : 07-2020]

model is also introduced with an accuracy of 56%. Despite this somewhat disappointing score, the authors claim that is still 2% higher than the average human lie-detection capabilities.

Text categorisation, on the other hand, is in itself a powerful and pragmatic approach against the spread of fake news. Researchers such as Badaskar et al. (2008) and Rubin et al. (2016) approach this problem by respectively training language models and TF-IDF weighted lexical vector-space models to classify the different types of news articles according to how they are written, with the latter system training also on satire and humorous fake news. An interesting alternative system for style detection is proposed by Afroz et al. (2012). The researchers base their study on the hypothesis that to a certain degree, linguistic features within a body of text change when respective authors try to hide their style of writing. By identifying these features, the authors point out that stylistic deception could be detected.

2.7.2 | Detection of Hyperpartisan News

Throughout this section, we examine different approaches implemented by other researchers with the hopes of addressing the spread of hyperpartisan news articles through ML classification systems. We first take a look at systems implementing traditional ML algorithms, before shifting to more complex DL solutions. Moreover, we tackle the addition of sentiment and model interpretability within the field of hyperpartisan news detection. In doing so, we compare and contrast published literature, analysing the performance, limitations, and overall results of the mentioned systems.

2.7.2.1 | Traditional Approaches

In classifying hyperpartisan news articles along with typical text-based binary classification tasks, one often finds Logistic Regression (LR) used as a baseline classifier with which other (more complex) algorithms are compared [Kiesel et al. (2019)]. Sengupta and Pedersen (2019) implement a LR classifier based on unigram features (Section 2.2.2.1) and a Convolutional Neural Network (CNN) DL classifier. Interestingly, the CNN's disappointing accuracy of 0.58 is exceeded by that of the LR model at 0.70, suggesting the possibility of the CNN model overfitting on the training data.

Alternatively, LR can be utilised as a feature extraction classifier. Working on the By-Publisher and the smaller By-Article SemEval datasets[13] (Section 2.1), Palić et al. (2019) train a LR model on the latter collection, to then be used on the former. All

[13] ***PAN SemEval Hyperpartisan News*** - `pan.webis.de/semeval19` [Last Accessed: 07-2020]

correctly labelled articles from the By-Publisher collection are then added to the smaller By-Article collection, thereby increasing the dimensions of the more reliable dataset.

Srivastava et al. (2019) manage to exploit a lot of the potential harnessed in an otherwise simple classification algorithm. The researchers make use of L2-regularised LR [Pedregosa et al. (2011)] - a variation of the LR classifier where the tuning parameter λ is defined during training on a validation set, or in the case of Srivastava et al. (2019), using 10-fold cross validation. They experiment with LR on Doc2vec, GloVe and Universal Sentence Encoder (USE) embeddings. The study achieves an accuracy score of 0.820 using USE embeddings, resulting in second place for the SemEval Hyperpartisan News Detection competition[14].

If the classification problem at hand is not linearly separable, more advanced alternatives to the LR may be required. The SVM is one such approach, capable of utilising a range of robust kernels, such as Linear, Polynomial, Sigmoid or RBF (refer to Section 2.3.1). Among a diversity of use cases, SVMs are widely used in NLP, along with the detection of hyperpartisan news articles [Alabdulkarim and Alhindi (2019); Cruz et al. (2019); Kiesel et al. (2019)]. Knauth (2019) implements two variations of an SVM model trained using an RBF kernel. The first is based on the stylistic features of the text corpus, while the second is based on content-related features. The two algorithms are evaluated against one another, with the former (based on the stylistic features) being the better performer.

Yeh et al. (2019) experiment with several approaches for the representation of article features. They evaluate a LR, a SVM classifier with linear kernel, and a SVM with RBF kernel on each of the representations. The authors note that the SVM exceeds LR at any given test, with the highest score being attributed to GloVe embedded feature vectors with an RBF-kernel SVM. A corresponding accuracy score of 0.796 is achieved on the training dataset, with an increase to 0.8057 (and F1 score of 0.7904) on the SemEval competition held-out test set, earning the system fourth place.

The SVM's popularity sees the classifier being used by a number of other systems with varying degrees of success [Kiesel et al. (2019)]. Systems like Isbister and Johansson (2019) evaluate the SVM (using a linear kernel) with several other traditional baseline classifiers and ULMFiT (Universal Language Model Fine-Tuning) - a DL classification model introduced by Howard and Ruder (2018). The SVM comes second to solely ULMFiT, with a respective accuracy of 0.7659 to that of 0.8025 in a fraction of the execution time. Moreover we see SVMs utilised with Word2Vec embeddings [Palić et al. (2019)],

[14] ***PAN SemEval Hyperpartisan News Detection (2019)*** - `pan.webis.de/semeval19` [Last Accessed: 07-2020]

word and character n-grams [Nguyen et al. (2019)], and sentiment features [Anthonio and Kloppenburg (2019); Palić et al. (2019)].

An interesting alternative to the two commonly used approaches above is Random Forests (RFs) - finding its way to the detection of hyperpartisan news articles [Kiesel et al. (2019)] as well. Cruz et al. (2019) take a feature-based approach to hyperpartisan news classification, experimenting with various algorithms including SVMs (with Squared Hinge-Loss), RFs, and Gradient-Boosted Trees (GBTs). What is known as Computer-Mediated Discourse Analysis (CMDA) [Herring (2004)] - an online researching approach for interactive behaviour, is used for acquiring details about the corpus such as type-token ratio and frequency of word n-Grams. The RF classifier provides the best classification performance with 100 estimators and the Gini split criterion. The model achieves the 10-fold cross validation accuracy of 0.763 on the By-Article dataset.

Alternatively, RFs are utilised with word-embeddings in order to exploit the context-based relations between vector embeddings and the quick, generalisable nature of the RF classifier. Systems like Stevanoski and Gievska (2019) combine the pretrained Word2Vec embeddings with RF. The authors also include various measures such as readability scores, stylistic and psycho-linguistic features, scoring a notable validation accuracy of 0.837 and a hidden test-set accuracy of 0.775.

Distinctively to the well-known traditional methods discussed above, one finds other, perhaps less well-known systems. Gupta et al. (2019) make use of XGBoost [Chen and Guestrin (2016)], a scalable end-to-end tree boosting system capable of scaling ML algorithms to use less resources. K-Nearest Neighbours (KNN), RFs, LR, and SVMs are also tested, with the best performer claimed to be KNN.

The MaxEnt (Maximum Entropy) [Ratnaparkhi (1996)] model is a statistical model trained on a corpus annotated with Part-Of-Speech (POS) tags. MaxEnt is used for the learning of supervised models by Agerri (2019) during the development of the *ixa-pipe-doc* system, a document classification system used on hyperpartisan news articles. *ixa-pipe-doc* is designed to be simple and efficient, combining various types of clustering features to provide denser document representations. Testing is performed on three types of features: 1) the current token, 2) the character n-grams of each token and 3) the word prefixes.

Throughout this section we have examined some of the most prominent traditional ML classifiers in use for the classification of hyperpartisan news articles. We started off by discussing the application of the LR classifier, a simple yet effective baseline model found among any ML task. This was followed by the SVM, arguably the most popular traditional algorithm found within the hyperpartisan news domain. An alternative ensemble model known as RF was also discussed, with our focus then shifting to other, less known systems - namely XGBoost and MaxEnt.

2.7.2.2 | Deep Learning Approaches

Differing from traditional ML techniques, the field of DL generally involves more complex and elaborate algorithms based on the Deep Artificial Neural Network (ANN) architecture. Throughout this section we examine published literature on the best performers in the area, namely CNNs, RNNs, and their improved alternatives LSTMs, for the detection of hyperpartisan news articles.

As of late, CNNs find themselves competing with RNNs for the best performing complex classification algorithm in the field of text-based NLP. One finds a good number of systems comparing the two approaches for the classification of hyperpartisan news, with outcomes differing according to each system's configuration [Kiesel et al. (2019)]. CNNs are also frequently compared to baseline traditional models such as Logistic Regression, SVM and Naive Bayes [Papadopoulou et al. (2019); Pérez-Almendros et al. (2019); Sengupta and Pedersen (2019)].

Zehe et al. (2019) make use of a CNN model working on Word2Vec [Mikolov et al. (2013a,b)] and FastText [Bojanowski et al. (2017)] word embeddings. Due to the natural setback of CNNs requiring features of the same size, all articles are represented with a fixed length of 2000 tokens. A base model CNN focusing on sentence level features is first implemented, later modified with the addition of a second convolutional layer in order to support article-level features. Finally, handcrafted features such as the number of tokens in the article and the average number of tokens per paragraph are appended to the flattened output of the pooling layers, to be integrated within the fully-connected layers of the CNN. The authors compare the CNN model with a linear SVM, achieving a highest, yet quite disappointing accuracy of 0.660 to the SVM'S 0.528 - using an ensemble model consisting of 5 CNNs with article-level support.

Jiang et al. (2019), coming first place in the SemEval Hyperpartisan News Detection challenge[15], exceed considerably the performance of Zehe et al. (2019). The authors use pre-trained ELMo embeddings [Peters et al. (2018)] based on word representations learned from character-level units. The article title is embedded preceding to the body. Inside of the CNN model, 5 convolutional layers are employed in order to cater for different kernel sizes. Batch normalisation is included for normalisation of the input distribution - reducing the possibility of overfitting and increasing the model's training speed. The researchers train 10 separate CNN models using 10-fold cross validation,

[15] ***PAN SemEval Hyperpartisan News Detection (2019)*** - `pan.webis.de/semeval19` [Last Accessed: 07-2020]

and choose the three best trained CNN instances to then organise into one ensemble model, where the average of the three models is considered as the final prediction. Experiments conducted produce a notable accuracy score of 0.84 during training and 0.82 during testing - a significant improvement on the poorer accuracy results reported by Zehe et al. (2019).

Other systems include evaluation of CNN with other niche systems such as Adaboost and SpaCy [Moreno et al. (2019)]. Indeed from the research we conducted, we conclude that the CNN is quite a popular algorithm of choice in the application of hyperpartisan news articles. One must not however ignore the strong competition brought forward by the RNN architecture - with promising literature employing both RNNs and LSTMs.

The two models are evaluated against one another by Zhang et al. (2019), where the researchers compare the performance of a CNN, a Recurrent CNN (RCNN), an LSTM and a biLSTM. Moreover an attention mechanism is integrated with the aim of determining textual features affecting the classification (more on this in Section 2.7.4). From the experiments conducted, an impressive highest test accuracy of 0.9368 is achieved using the biLSTM architecture with added attention. The other models are not far off, with the LSTM and CNN getting the same range of accuracies (0.9174 and 0.9147). Despite these very high scores, the submitted models struggle on the SemEval hidden test set, with the corresponding accuracy dropping to 0.683 on the By-Article set and 0.652 on the By-Publisher. This behaviour strongly suggests the possibility of model overfitting, especially since such high validation accuracies are not shared by systems performing significantly better on the hidden test set.

Cramerus and Scheffler (2019) classify hyperpartisan news articles by using an LSTM to capture the relationships between textual features. Article features are represented by word embeddings generated through a custom built Skipgram Word2Vec model [Mikolov et al. (2013b)]. Three LSTM models are trained, one on the By-Publisher hyperpartisan training set, one on the By-Publisher hyperpartisan validation set, and one on the separate, By-Article collection. The correctly predicted articles are then passed on to the final model - a separate LSTM trained on the newly chosen articles. A somewhat disappointing accuracy of 0.652 is acquired, however as the researchers themselves claim, the solution performs considerably better on the less refined By-Publisher hyperpartisan dataset (refer to Section 2.1). Hence if otherwise trained on the more reliable By-Article data collection, better results would have been probable.

The CNN and RNN neural networks do not strictly have to be used separately. Papadopoulou et al. (2019) join the two by using a CNN along with a biRNN. The CNN is utilised first, capturing the word sequence structure at different levels of granularity. Following the CNN is a modified version of the biRNN known as Bidirectional Gated Recurrent Units (biGRU) - calculating the sentence vectors by taking into account the context formed by their neighbouring sentences. Despite a poor outcome of 0.575 accuracy on the hidden test set, the researchers are positive that a fusion solution determining the best classifier to use between traditional and DL methods could theoretically raise this score to 0.85.

The disruptive performance promised by the Transformer neural network architecture finds itself introduced as well to the problem of hyperpartisan news classification. BERT (Section 2.2.2.2) is a specialised transformer model presented by Devlin et al. (2019), capable of generating several layers of context-dependent word embeddings for any textual input feature. The generated embeddings could then either be extracted and used as desired, or classified internally within the model itself by the addition of a classification layer.

Hence one finds BERT Embeddings used in conjunction with Linear [Hanawa et al. (2019)] and Softmax [Mutlu et al. (2019)] classifiers. Being the smallest and most versatile BERT pretrained model, BERT-Base is used by systems like Ning et al. (2019) and Lee et al. (2019). Interestingly, Lee et al. (2019) evaluate the implementation of 2 LSTMs (one for the article title and one for the body) with a multilayer perceptron classifier appended with the BERT transformer. Despite initially performing better, the LSTMs are exceeded by BERT (accuracy of 0.758, F1 of 0.7647) once pseudo-label de-noising is applied as a preprocessing step on the dataset.

2.7.3 | Sentiment in Hyperpartisan News Articles

Given the sensationalised nature of hyperpartisan news articles, one may hypothesise that sentiment plays a pivotal role within the opinionated texts encompassed by such articles. Thereby, in designing tools for the detection of such news, we find a good number of studies investigating the role of sentiment within hyperpartisan articles.

The NLTK VADER [Hutto and Gilbert (2014)] is perhaps one of the most in-use and popular out-of-the-box sentiment labelling solutions. Due to its popularity one finds it embedded within a number of hyperpartisan news detection systems [Kiesel et al. (2019)].

One such system, which also stands out due to its unique approach, is introduced by Anthonio and Kloppenburg (2019). The researchers approach the problem of hyperpartisan news detection in a novel way where the classification is based solely on the sentiment features of the article. Albeit resulting in unimpressive scores both in terms of accuracy (0.5616) and F1 score (0.5717), the approach is definitely interesting and the authors do manage to increase the accuracy score by around 13% from 0.4310 to 0.5616 when using negative VADER sentiment as opposed to compound (Section 2.4.2).

Palić et al. (2019) make use of several external features along with sentiment, among which are the publication date and the number of quotations within the article. Different to the system above, the authors include all four sentiment scores provided by VADER (positive, negative, neutral and compound), consequently reporting an increase in accuracy from 0.75663 to 0.76128.

A similar approach is taken by Joo and Hwang (2019), with multiple feature groups being supplied to the classification process. Despite achieving high overall results, performance seems to suffer with sentiment features, which when tested individually, resulted in an accuracy of 0.6124 and an F1 of 0.61. Interesting to add that this is a different experience to the one reported by the preceding two systems, yet one that in conducting our experimentations we found ourselves agreeing with, as discussed further in Chapter 4.

Alternatively to the popularity of VADER, one does find a minority of systems utilising other lexical-based sentiment engines. Amason et al. (2019) employ two relatively smaller lexicons, with the first containing 2000 words of positive sentiment, and the second containing 4000 labelled as negative. Chen et al. (2019) on the other hand make use of the TextBlob[16] Python library - performing sentiment analysis on the articles and their titles. Using a Naive Bayes classifier, the authors report improvements, just as same as Anthonio and Kloppenburg (2019) and Palić et al. (2019), when sentiment features are included.

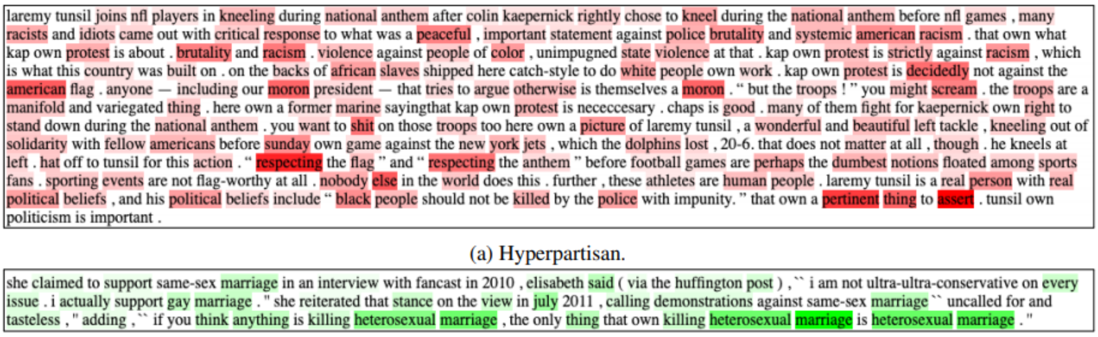
2.7.4 | Model Interpretability and Saliency of Features

The interpretation or explainability of a ML model is arguably still an emerging discipline within the realm of A.I. Moreover, one may not directly realise the importance of such an addition to a typical classification system. In the problem of hyperpartisan news detection, we feel that such functionality provides a double benefit to the study; an insight into the reasoning of the classifier in performing its predictions, and perhaps more importantly, the nature of features within an article pivotal to the hyperpartisan classification. Despite the infantile stages of such a niche area, we are encouraged by two similar systems employing preliminary work on defining the features behind the model's prediction.

Zhang et al. (2019) employ a Bidirectional LSTM (biLSTM) neural network with the addition of a self-attention mechanism [Zhou et al. (2016)]. The attention mechanism

[16] ***TextBlob*** - `textblob.readthedocs.io` [Last Accessed : 07-2020]

is applied on the output vector produced by the biLSTM, with a simple visualisation (featured in Figure 2.9) showcasing the article features and their corresponding attention scores. Through this system the authors note that heavily polarised, negative words such as *moron* and *racist* are often associated with hyperpartisan articles.



laremy tunsil joins nfl players in kneeling during national anthem after colin kaepernick rightly chose to kneel during the national anthem before nfl games , many racists and idiots came out with critical response to what was a peaceful , important statement against police brutality and systemic american racism . that own what kap own protest is about . brutality and racism . violence against people of color , unimpugned state violence at that . kap own protest is strictly against racism , which is what this country was built on . on the backs of african slaves shipped here catch-style to do white people own work . kap own protest is decidedly not against the american flag . anyone — including our moron president — that tries to argue otherwise is themselves a moron . " but the troops ! " you might scream . the troops are a manifold and variegated thing . here own a former marine sayingthat kap own protest is nececcesary . chaps is good . many of them fight for kaepernick own right to stand down during the national anthem . you want to shit on those troops too here own a picture of laremy tunsil , a wonderful and beautiful left tackle , kneeling out of solidarity with fellow americans before sunday own game against the new york jets , which the dolphins lost , 20-6. that does not matter at all , though . he kneels at left . hat off to tunsil for this action . " respecting the flag " and " respecting the anthem " before football games are perhaps the dumbest notions floated among sports fans . sporting events are not flag-worthy at all . nobody else in the world does this . further , these athletes are human people . laremy tunsil is a real person with real political beliefs , and his political beliefs include " black people should not be killed by the police with impunity. " that own a pertinent thing to assert . tunsil own politicism is important .

(a) Hyperpartisan.

she claimed to support same-sex marriage in an interview with fancast in 2010 , elisabeth said (via the huffington post) , `` i am not ultra-ultra-conservative on every issue . i actually support gay marriage . " she reiterated that stance on the view in july 2011 , calling demonstrations against same-sex marriage `` uncalled for and tasteless , " adding , `` if you think anything is killing heterosexual marriage , the only thing that own killing heterosexual marriage is heterosexual marriage . "

(b) Non-hyperpartisan.

Figure 2.9: A visualisation of feature saliency within an article flagged by a biLSTM Self-Attention Mechanism. Note that the colour intensity corresponds to the saliency towards or against a prediction class. Source: Zhang et al. (2019).

Amason et al. (2019) address similar issues through a completely different, yet equally interesting angle. Having generated a large number of both internal and external features (including article title, BoW features, sentiment and complexity features), the researchers employ a feature selection approach based on the Chi-Squared [Greenwood and Nikulin (1996)] statistical test. They attempt to establish the most distinguishing features, and in doing so, assigning each feature a corresponding score, with the 10 most influential features along the whole training set included in the authors' published literature. One finds words such as *trump, political, israel...* within this list.

2.8 | Background and Literature Review - Summary

Throughout this chapter we discussed a background overview of the techniques implemented in the HyperPT system for the classification of hyperpartisan news articles. We started off in Section 2.1 by discussing the SemEval Hyperpartisan News Articles dataset - the collection of data upon which the study is performed. We then advanced to Section 2.2 and Section 2.3, where we respectively examined the feature preprocessing, representation and classifiers employed within our system - thereby completing the basic classification pipeline.

In Section 2.4 we provided a brief background on the addition of sentiment, before repeating the process for model explainability in Section 2.5. Evaluation criteria is discussed in Section 2.6, to finally, in Section 2.7, examine similar systems employing the same or alternative methods for the classification of fake news (a neighbouring problem) and hyperpartisan news.

In Chapter 3 we now discuss our design process in building the HyperPT system. In doing so we refer back to the background and related work discussed throughout this chapter as a means of guidance and further information on the proposed methodology.

3

System Methodology

With a thorough background on the problem of hyperpartisan news addressed in Chapter 1, along with a concise review of the related work in Chapter 2, we now discuss the design process behind the implementation of the HyperPT system.

In Section 3.1 we first discuss the HyperPT system from a high level overview, before moving on to Section 3.2, where we address the extraction, preparation and preprocessing of data. The implementation of various feature representation techniques is examined, before the corresponding classification algorithms are integrated within the system as highlighted in Section 3.3. Finally, we address the addition of external features, namely sentiment, model explainability and consequently feature saliency, featured in Section 3.4.

3.1 | System Overview

In building the HyperPT system, we base our implementation on the concept of modularity - where components can be added or discarded from the main pipeline with relative ease. This enables us to modify and extend the project implementation relatively quickly, allowing us to quickly adapt its configurations to accommodate various test scenarios. To accomplish this, we create separate components, each responsible for a specific project functionality, communicating and sharing data with one another through a main workflow pipeline.

HyperPT can be summarised to three components - as showcased in Figure 3.1. The first component is the data loading and preparation, where data is loaded from file, preprocessed, cleaned and converted into the chosen feature representation vectors. The second component handles the classifiers, where we initialise our chosen classifier and pass the transformed data samples to it for classification. As discussed further on in

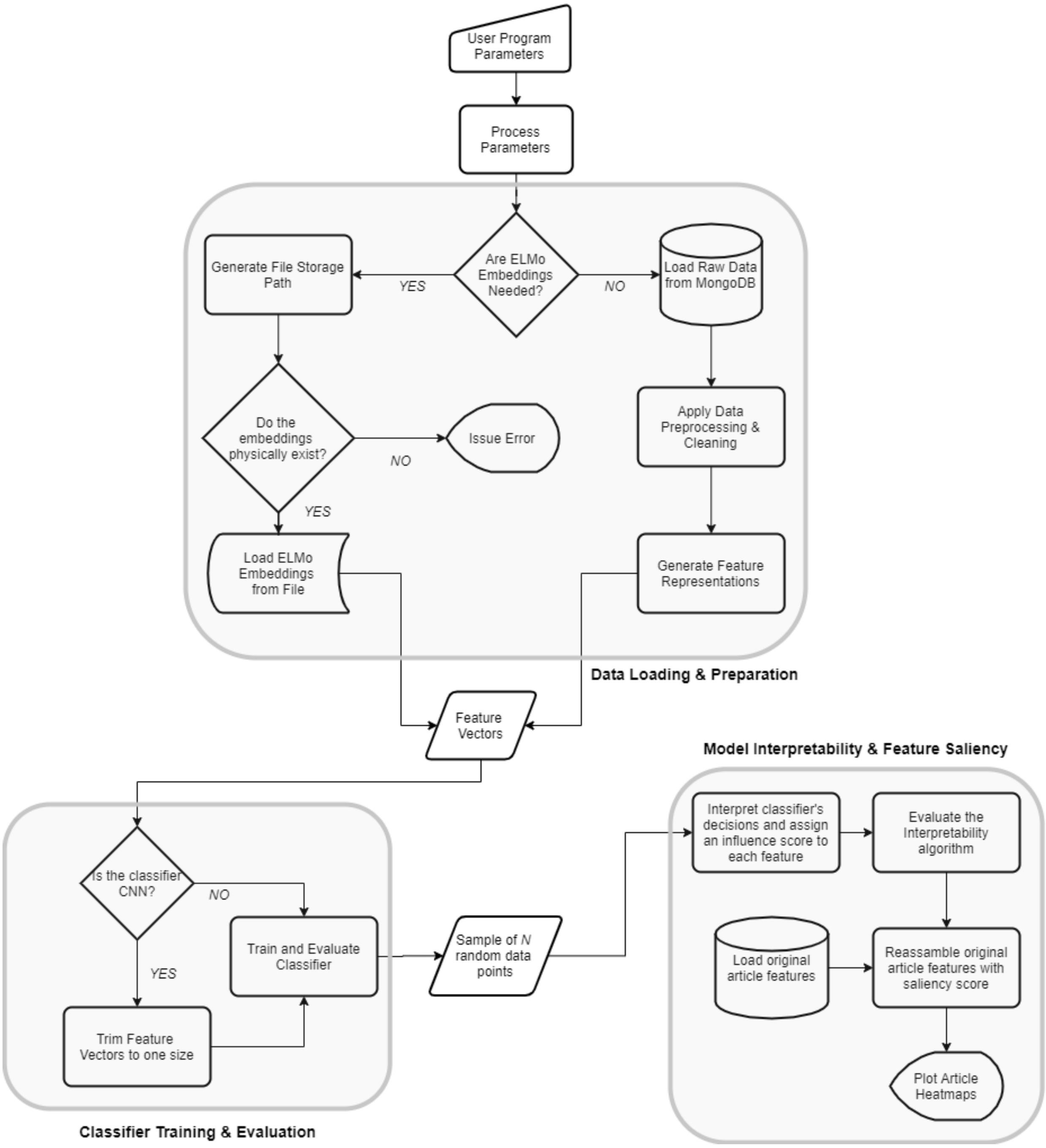

User Program Parameters
Process Parameters
Are ELMo Embeddings Needed?
YES
NO
Generate File Storage Path
Load Raw Data from MongoDB
Do the embeddings physically exist?
NO
Issue Error
YES
Apply Data Preprocessing & Cleaning
Load ELMo Embeddings from File
Generate Feature Representations
Data Loading & Preparation
Feature Vectors
Model Interpretability & Feature Saliency
Is the classifier CNN?
NO
YES
Train and Evaluate Classifier
Trim Feature Vectors to one size
Sample of N random data points
Interpret classifier's decisions and assign an influence score to each feature
Evaluate the Interpretability algorithm
Load original article features
Reassamble original article features with saliency score
Plot Article Heatmaps
Classifier Training & Evaluation

Figure 3.1: A high-level visualisation of the HyperPT design process. The system is made up of three sequential main components; the Data Loading & Preparation, Classifier Training & Evaluation, and finally Model Interpretability & Feature Saliency.

Section 3.3, a number of optional parameters can be included at this stage to define the classifier's execution and evaluation.

The third and final component is the model interpretation where, once the model has been trained and evaluated, we perform analysis on its decision-making by employing explainability algorithms discussed back in Section 2.5. Moreover, at this stage we also perform article reconstruction - taking into consideration the saliency of each feature assigned through the model explainability.

The HyperPT workflow is highly configurable through command-line arguments, using which we instantly inform it which data to load, what preprocessing to apply, how to represent the data features, and with which classifier to process them with (among other configurations and customisations). Such functionality allows us to not only perform specific tests on the fly, but also run multiple tests in bulk through batch execution tasks.

Using these three components we establish the complete workflow of the HyperPT system. One must also consider three other, ad hoc subcomponents - used either independently as preprocessing steps or by the main functionality blocks addressed above. The first is the I/O Bridge, handling all communication and data flow with databases (MongoDB) and storage. Second is the Word Embedding Generator (Section 3.2), which generates feature word embedding vectors and saves them to file storage (through the I/O Bridge) for fast ad hoc usage. Third and final subcomponent is the Sentiment Generator. This component is run independently, assigning sentiment labels according to the user preferences to the given data collection (Section 3.4.1).

The whole system is built using Python 3.6.10[1]. Two Anaconda[2] (v4.8.3) environments are set up, with the main one used by the three system components and based on Tensorflow[3] 1.13.1 (with GPU support), and the other based on Pytorch[4] 1.4.0 (with GPU support) used by the Word Embedding Generator and Sentiment Generator subcomponents. Moreover, MongoDB[5] 4.2.1 is utilised as the primary data loading source in which the parsed dataset is initially imported, and from which data samples are loaded during runtime.

We feel that the technology used is of significant contribution to the extents we go to in terms of system design and implementation. With a number of different technologies, approaches and components working together as one coherent system, the implementation of such modules is not trivial. This however, is achieved through the flexibility and

[1] ***Python 3.6.10*** - `www.python.org` [Last Accessed : 07-2020]

[2] ***Anaconda*** - `www.anaconda.com` [Last Accessed : 07-2020]

[3] ***Tensorflow*** - `www.tensorflow.org` [Last Accessed : 07-2020]

[4] ***Pytorch*** - `pytorch.org` [Last Accessed : 07-2020]

[5] ***MongoDB*** - `www.mongodb.com` [Last Accessed : 07-2020]

compatibility offered by the Python programming language, along with the simplified importation of crucial libraries through Anaconda and the quick and direct data I/O through MongoDB. Moreover one must also mention the pivotal contribution given by the Tensforflow, Pytorch, Keras[6] (v2.2.4) and Scikit-Learn[7] (v0.22) libraries, all of which are crucial in the precise and efficient building of any and all classification algorithms used.

The project is developed and run on a Windows 10 System, with 32GB of RAM and an AMD Ryzen 7 3700x 8-Core Processor running at 3.6GHz. Moreover, an NVIDIA RTX2080 is utilised as the GPU unit, proving crucial for the fast generation of word embeddings, execution of Deep Learning classifications, and overall system evaluation. In replicating the system, the hardware highlighted above is not a requirement, yet is a strong recommendation for better adaptation to the system implementation and faster execution time.

3.2 | Data Loading, Preprocessing and Representation

The SemEval Hyperpartisan News Article dataset, assembled by Kiesel et al. (2019) consists of two collections; the smaller By-Article collection amounting to 645 articles, and a larger By-Publisher collection consisting of $600,000$ articles. As discussed back in Section 2.1, we make use of solely the By-Article collection due to its superior quality of labelling - thereby resulting in a more stable model evaluation process. We make use of the provided labels highlighting whether an article is hyperpartisan or not in order to build our classification system.

Having acquired the SemEval Hyperpartisan News Article dataset in XML form, we initially extract the original XML and write the data samples to a MongoDB database. Being a document-based system, data is represented with each database document being equivalent to an article data sample. The corresponding attributes are then added, including the article hyperpartisan label. We decide on using MongoDB as the main data source due to its quick and flexible extraction of data through the PyMongo Python library. Moreover the I/O Bridge subcomponent is designed around this library such that I/O requests are abstracted by simple, efficient functions.

[6] ***Keras*** - `keras.io` [Last Accessed : 07-2020]
[7] ***Scikit-Learn*** - `scikit-learn.org` [Last Accessed : 07-2020]

3.2.1 | Data Preprocessing

Data is fetched during runtime, with data extracted being in raw (normal) form. Ad hoc preprocessing is then performed before it is transformed into corresponding feature representations. Based on our research of similar systems conducted in Chapter 2, we choose four individual preprocessing filters - the removal of stopwords, removal of punctuation, feature lowercasing and feature lemmatisation. Stemming is left out of the system due to the more promising approach of feature lemmatisation (Section 2.2.1). During system evaluation, the best performing three techniques are also applied aggregately on the corpus, amounting to four other preprocessing combinations. In all, considering raw unedited features, individual and aggregate data preprocessing, we are left with 9 separate preprocessing approaches. Following this data cleaning process, the article string is split into individual words in a simple yet crucial process called tokenization.

3.2.2 | Feature Representation

The preprocessed and tokenised data is at this stage ready to be transformed into the feature representation of choice. In deciding on which feature representations to consider (the same as with classification techniques as discussed in Section 3.3), we review the sizeable amount of possible representation methods in use by related systems (refer to Section 2.7), along with novel approaches which may have not yet been explored.

When it comes to traditional feature representation approaches, we settle on the most well-known, effective techniques which have stood the test of time across multiple research domains within NLP - including the detection of hyperpartisan news [Chen et al. (2019); Joo and Hwang (2019); Knauth (2019)]. Being the most straight-forward and simple of the lot, we first consider the Term-Frequency (TF) Bag-of-Words (BoW) representation. As a well-known promising variation on this approach due to the addition of weights to the input features, we also utilise the Term Frequency - Inverse Document Frequency (TF-IDF) measure. Finally, we experiment with POS-Tags, due to the generalised feature labelling provided by such a technique.

The newly emerging yet heavily researched field of word embeddings promises notable improvements on classical feature representation approaches [Goldberg and Levy (2014); Lavelli et al. (2004)]. Moreover the years of 2018 and 2019 saw the rise of deep contextualised word embeddings such as ELMo (Embeddings from Language Models) [Peters et al. (2018)] and BERT (Bidirectional Encoder Representations from Transformers) [Devlin et al. (2019)], offering more detailed multi-layered word embeddings generated from the features themselves and surrounding contexts.

Analysing published literature, particularly projects submitted to the SemEval 2019 Hyperpartisan News Article Detection challenge (Section 2.1), we decide to include both standard word embedding systems, namely Word2Vec and GloVe, and deep contextualised representations that are ELMo. In doing so we can examine the differences in performance (if any) encountered between the two approaches, and whether these bring significant change. Note that since as described in Section 4.1.1, ELMo's original output consists of three vectors, we adapt the approach utilised by Jiang et al. (2019) and average the three outputs into one vector.

Along with these three approaches, we initially considered as well the addition of BERT, eventually deciding against it. This decision was reached due to various reasons. First of all, in perusing related systems we noticed that BERT is utilised by a number of systems [Drissi et al. (2019); Lee et al. (2019); Ning et al. (2019); Shaprin et al. (2019)] with notably different ranges of success. On the other hand, ELMo Embeddings (to the extent of our research) are solely used by one system; Jiang et al. (2019), which also happens to be the winner of the SemEval Hyperpartisan challenge, boasting high performance results.

Moreover as discussed further below, generating the large number of deep contextualised embeddings needed to cover all test scenarios consumed a large amount of time. Hence due to the limited time allocated to this study, we decided to safely perform complete system implementation and evaluation using the three technologies mentioned above, opting to leave out BERT due to the further complexity which would have otherwise been added to the implementation. This in the end was the right call to make, since building the system, its evaluation and the corresponding dissertation did take a serious amount of time to conclude. Increasing the complexity and volume of this work would have run us the otherwise high risk of rendering the project incomplete. We nevertheless consider the addition of BERT as future work, which we think would make for an interesting extension to the HyperPT system.

All of the utilised embedding technologies are pretrained. The Word2Vec model is used twice, both of which are pretrained on the Google News dataset (consisting of around 100 billion words), while one being further retrained at runtime on our hyperpartisan news dataset. GloVe on the other hand is pretrained on the Common Crawl[8] 42 billion token dataset (with other configurations also available). Lastly, ELMo comes pretrained on the 1 Billion Word Benchmark [Chelba et al. (2013)] - a benchmark corpus for the monitoring of statistical language modelling. Note that while both Word2Vec

[8] ***Common Crawl*** - `commoncrawl.org` [Last Accessed : 07-2020]

and GloVe models are implemented using the Gensim[9] library, ELMo is implemented using AllenNLP[10].

As mentioned above, initial word embedding generation is performed during runtime, such that data samples are loaded from MongoDB, cleaned and converted to word embeddings. This is not a feasible approach when it comes to ELMo, with just the generation of embeddings taking the vast majority of the whole runtime. To address this, we design a simple yet pivotal file system handler, which translates the embedding parameter given as a command-line variable during execution to the specific directory containing the equivalent ELMo embeddings. This allows us to generate word embeddings as a one-time preprocessing task, saving them to a unique sub-directory as HDF5 binary files - in doing so reducing the runtime loading of ELMo embeddings from over 30 minutes to around 20 to 30 seconds.

3.3 | Classification Approaches

Having reviewed related work discussed in Chapter 2, we employ within HyperPT both traditional and Deep Learning (DL) classification approaches. This is since one must consider that despite the more elaborate architecture presented by DL classifiers, traditional approaches are 1) still very much in use and 2) provide different and unique benefits to DL approaches. This is apparent in similar work discussed in Section 2.7, where related systems [Isbister and Johansson (2019); Sengupta and Pedersen (2019)] report high accuracies using traditional methods and close performance results between traditional and DL approaches.

3.3.1 | Traditional ML Approaches

The Support Vector Machine (SVM) is one of the most well known traditional classification models employed within the classification of hyperpartisan news. It is often found compared to both traditional and DL models. It is capable of not only matching the performance of baseline algorithms like the Logistic Regression (LR), but also improving on it. Further to this, a number of kernels are available to choose from depending on the nature and complexity of the classification problem.

As we have seen in Section 2.3, the SVM is a popular and effective classifier within ML and the domain of hyperpartisan news detection. Moreover we notice how similar systems [Palić et al. (2019); Yeh et al. (2019)] adapt word embeddings to fit this model -

[9] ***Gensim*** - `radimrehurek.com/gensim` [Last Accessed : 07-2020]

[10] ***AllenNLP*** - `allennlp.org/elmo` [Last Accessed : 07-2020]

an approach we also pursue in order to evaluate the potential of word embeddings not only with DL approaches but also with traditional classification models. Backed by this reasoning, we settle on the SVM as the first model to employ for the classification of hyperpartisan news articles. Despite the majority of related work making use of either linear or RBF kernel SVMs, we evaluate also the Sigmoid and the Polynomial kernels, establishing the best system through 10-fold cross validation and hyperparameter tuning - discussed in detail in Section 4.2.1.

The Random Forest (RF) classifier is a popular alternative approach proposing unique advantages not inherently found in other methods such as the SVM. Indeed as discussed in Section 2.3, the RF is in itself an ensemble model of a number of decision trees. It is capable of easily adapting to non-linear problems, and naturally resists the tendency to overfit on the training data. In considering this model, we strongly focus on this property, since in case other models would be getting overfitted on our training data, RF would act as the baseline with which the degree of overfitting could be measured and corrected. Moreover, we see word embedding technologies adapted to the RF classifier as well, with systems such as Stevanoski and Gievska (2019) combining pretrained Word2Vec embeddings with the classifier. One also finds the neighbouring classifier Gradient-Boosting Trees (GBTs) employed in the domain of hyperpartisan news. Systems such as Cruz et al. (2019) compare this method and its performance with that of the RF - with RF being the better performer. We hence integrate the RF classifier as the second contestant for the classification of hyperpartisan news.

Given the generalisation and adaptability capabilities of the SVM through its range of kernel functions, and the unique advantages brought forward by the RF such as the ensemble of decision trees and rejection to overfitting, we conclude on these two classifiers as candidates for the classification of hyperpartisan news articles. We hence compare the performance of these classifiers with one another, and with selected DL systems, as detailed in Section 3.3.2.

3.3.2 | Deep Learning Approaches

Having established the traditional classification methods to employ, we examine state-of-the-art DL classification approaches for the classification of text-based NLP tasks and, more specifically, the classification of hyperpartisan news articles. This space is dominated by CNNs and RNNs - or rather their optimised relative, the LSTM networks.

One finds varying performances with both these algorithms. In evaluating similar systems employing such techniques, we notice a minority of studies [Färber et al. (2019); Zhang et al. (2019)] achieving high validation accuracies on the local training dataset, yet suffering when tested on a hidden test set (refer to Section 2.3). This behaviour strongly suggests the tendency of overfitting on the training data, hindering the model's capability to generalise to unseen data. Such an issue is an added difficulty to our analysis, since the resulting poor performance of a model may not be due to the model's lack of ability but rather the implementation in which it is used.

Jiang et al. (2019), coming first place in the SemEval Hyperpartisan News Detection challenge, successfully implement a CNN classifier on the By-Article Hyperpartisan dataset (Section 2.1). Moreover, after training 10 separate classifiers using 10-fold cross validation, the best three are handpicked and aggregated within an ensemble system in which the output of all three is averaged into one. Each of the classifiers makes use of five convolutional layers, while batch normalisation is utilised within the classifier for added performance and faster training time. We choose to employ the same individual CNN architecture due to its reputable performance and generalisation capabilities showcased when tested on the hidden test-set. Moreover we also experiment with the ensemble model suggested by Jiang et al. (2019), as discussed in detail in Section 4.4.

Given the time constraints imposed upon the project, we include solely one DL classifier within HyperPT - the CNN. Our decision on employing the CNN network is largely based on the success displayed by Jiang et al. (2019) - it is not however the only reason. Compared with the RNN network, the CNN trains significantly faster (up to ×5, as shown by DeepBranch[11] DL benchmarking tools). Moreover the RNN and LSTM network family contains a number of various architectures, and are often utilised in double, opposite formations known as biRNNs or biLSTMs. This is such that equal attention is given to the whole sequence features (Section 2.3). Given this reality, such a network would have been more challenging and time-consuming to train and evaluate in the given timeframe - more so when considering the seemingly overall poorer performance featured in related work (Section 2.7), with the highest rank achieved using such models being fifth place [Isbister and Johansson (2019)]. Nevertheless we do not exclude that further research into the RNN architecture is possible, and given the opportunity to do so as future work, the addition of RNN and LSTM networks and their evaluation with the already implemented models would be an important extension to the HyperPT system.

[11] ***DeepBranch*** - `github.com` [Last Accessed : 07-2020]

3.3.3 | System Implementation and Hyperparameter Tuning

Further to the above, we are left with three classification algorithms; SVM, RF and CNN. The former two algorithms are both implemented using the Scikit-Learn[12] Python library, while the CNN is implemented using Keras[13] with a Tensorflow[14] backend. Within the physical system implementation, each classifier is added as a separate class object module, with the option of which one to execute during runtime specified as a command-line parameter. Each classifier can be evaluated in two configurations; 10-fold cross validation or train-test split (with the test set being either randomly selected or statically pre-chosen). Both cross validation and train-test splitting is performed using the Scikit-Learn framework, making use of Scikit-Learn Keras wrappers[15] for supporting the CNN implementation.

Hyperparameter tuning is performed on all three classifiers as discussed thoroughly in Section 4.2.1. In doing so we attempt at finding the best classifier configuration for the problem at hand, adapting the models as much as possible. Tuning is performed in a grid-search and cross validation manner, automated using the Scikit-Learn Grid-SearchCV model selection approach, which couples the two techniques automatically.

3.3.4 | Adapting Feature Representations to the Classifier Architectures

The CNN classifier presents the inherent requirement that all data sequences given as input must be of the same length. Thereby we preprocess each article feature length such that they all contain the same, pre-set number of features. Hence in the case that an article is longer it is trimmed to fit, while articles of smaller lengths are appended with zero vectors. Through initial tests we determined the best pre-set threshold for each feature representation approach, settling on 500 features for Word2Vec, 300 features for GloVe and 1024 features for ELMo. Note that traditional feature representation techniques are of equal length already.

Moreover, evaluating Word2Vec features results in better performance using the Skipgram model, which is thereby maintained throughout ensuing tests involving Word2Vec embeddings.

The adaptation of word-embedding features for the traditional classifiers (namely the SVM and RF) involves transforming the input features from 2-dimensional vectors

[12] ***Scikit-Learn*** - `scikit-learn.org` [Last Accessed : 07-2020]

[13] ***Keras*** - `keras.io` [Last Accessed : 07-2020]

[14] ***Tensorflow*** - `www.tensorflow.org` [Last Accessed : 07-2020]

[15] ***Scikit-Learn Wrapper for Keras*** - `pypi.org` [Last Accessed : 07-2020]

(vector for each feature) to one vector representing the entire article. Analysing related work, similar systems tend to average all article feature vectors into one [Srivastava et al. (2019); Yeh et al. (2019)]. We experiment with a more niche approach featured by De Boom et al. (2016) as a baseline method, in which we take the element-wise minimums and element-wise maximums of all the feature vectors. These two vectors are then appended, with the minimums extended by the maximums - resulting in a single vector of double the length of the original.

Initial evaluations of this approach with more basic ones such as the element-wise mean, minimum and maximum resulted in min-max reduction achieving the best overall performance. Alternative approaches such as max-min reduction are not attempted due to time-constraints, however we do recommend the consideration of these approaches in future experiments. Alternatively, approaches such as Principal Component Analysis (PCA) [Abdi and Williams (2010); Wold et al. (1987)] could be used to reduce the dimensionality of the feature vectors by deriving their principal components.

Having discussed the selection and implementation of the three classification candidates for the detection of hyperpartisan news articles, we now move to Section 3.4, where we shift our focus to external features, particularly the addition of sentiment and model interpretation along with the corresponding feature saliency.

3.4 | External Features

In Section 3.2 and Section 3.3 we discussed our approach and implementation of the hyperpartisan classification system, including the loading, preprocessing and representation of data samples, along with the three classifiers used; SVM, RF and CNN.

In this section we now examine the addition of external features, with the aim of increasing the accuracy and reliability of our classification performance, while giving us better insight into the nature of the hyperpartisan news article itself (as discussed later on in Chapter 4). First in Section 3.4.1 we discuss the integration of sentiment features within HyperPT - highlighting the various forms of sentiment applied. Following this, in Section 3.4.2 we describe our approach towards implementing an Explainable A.I. system capable of interpreting the classifier’s decisions and consequently highlighting the saliency of each article feature within the classification.

3.4.1 | Sentiment Features for Hyperpartisan News Classification

Due to the potentially crucial role sentiment may play in sensationalised written text such as hyperpartisan news articles, we feel that it is one the most pivotal external features to consider within the classification. Thereby, further to the concise discussion on its underlying functionality in Section 2.4, we use the NLTK VADER (Valence Aware Dictionary for sEntiment Reasoning) sentiment lexicon to issue sentiment labelling for our corpus. Being proved by Hutto and Gilbert (2014) as one of the top off-the-shelf approaches currently in use, one finds VADER utilised within the application of hyperpartisan news detection [Anthonio and Kloppenburg (2019); Joo and Hwang (2019)]. We opt for this method not only due to its reliable sentiment labelling, but also because of its significantly faster classification time compared with other, more complex methods.

Sentiment features are generated as a one-time process by a separate subcomponent of the HyperPT system known as the Sentiment Generator. The corresponding sentiment scores are then written as attributes to the respective articles inside of the MongoDB database, to be loaded accordingly at runtime. During classification, sentiment scores are embedded within the article text, preceding the corresponding textual features.

We make use of sentiment in two forms; label and score. A sentiment label is generated by rounding the original sentiment score, with high sentiment scores (larger than 0) being classified as positive and low sentiment scores (smaller than 0) being classified as negative. A sentiment score of 0 implies that the text is of neutral sentiment. Sentiment scores issued by VADER are four; positive, negative, neutral and compound. The compound sentiment score is a float number between -1 and 1, with -1 being extreme negative, and 1 extreme positive. In calculating this measure, VADER takes into consideration the valence of each feature with respect to the rest of the corpus in which it is used - thereby giving us a more true-to-context sentiment measure [Hutto and Gilbert (2014)].

We implement sentiment at two levels of granularity; at the article level and the sentence level. These result in a single sentiment feature appended in front of the article text in case of the former, and a localised sentiment feature appended in front of each sentence sequence for the latter. At the article level we generate the sentiment label using two separate approaches; Global Label - where we calculate one global article sentiment score, and Derived Label, which is calculated by taking the average of all the sentiment scores generated at the sentence level, thereby in theory giving us a more locally-aware measure.

Among the number of presented approaches, we also take into consideration just the generated negative score, as well as all VADER scores altogether (Positive, Negative, Neutral, Compound). We do so inspired by related work, in order to determine whether these approaches improve on the system performance as respectively reported by similar systems - Anthonio and Kloppenburg (2019) and Palić et al. (2019) (Section

1	Global Label - Article Level
2	Derived Label - Article Level
3	Compound Score - Article Level
4	Compound Score scaled ×1000 - Article Level
5	Negative Score - Article Level
6	Negative Score ×1000 - Article Level
7	Label - Sentence Level
8	Compound Score - Sentence Level
9	Compound Score ×1000 - Sentence Level
10	Negative Score - Sentence Level
11	Negative Score ×1000 - Sentence Level
12	Positive, Negative, Neutral, Compound - Article Level
13	Positive, Negative, Neutral, Compound ×1000 - Article Level

Table 3.1: Sentiment Integration Approaches employed at the article and the sentence level within the HyperPT study.

2.4). Moreover we scale each sentiment score separately by a factor of 1000, to determine the effects of such an enhancement (note that during initial testing we also scaled the features by 10 and 100, with close to no difference in performance). In all we employ 13 different sentiment integration approaches, as showcased in Table 3.1.

3.4.2 | Explainable A.I. for Model Interpretation and Feature Saliency

The newly emerging field of Explainable A.I. allows one to determine the logic behind a classifier's decision, in doing so indicating whether it is working as expected. Moreover, it gives one further insight into the data samples and the effects the features play within the classification.

Throughout the HyperPT study we make use of a novel effective Explainable A.I. algorithm known as LRP. This approach, as discussed thoroughly in Section 2.5, propagates back through the classifier, analysing each layer of the model until it assigns an influence score to each input feature. Moreover we utilise a baseline explainability approach known as Sensitivity Analysis (SA) with which to compare the LRP algorithm.

LRP is put to use once the training and evaluation of the classification model is concluded. It is run on the trained classifier, monitoring N randomly chosen articles. Due to the assigning of influence scores to each input feature, a saliency map for all the article features is created, through which we determine which features are the most relevant to the classification. Hence say if a hypothetical topic of classification would be space exploration, words like *astronauts* and *NASA* would have more saliency in the classification decision compared with other words which are not as directly related to the subject - such as *people* and *environment* [Arras et al. (2017)].

Before we analyse the features and corresponding influence scores, we make sure to evaluate the explainability algorithm itself to be sure that it is trustworthy in its predictions. LRP is evaluated using a technique inspired by Arras et al. (2017) and Samek et al. (2017), whereby we sequentially remove the saliency-flagged features in descending order and reclassify the corresponding article samples. In doing so we monitor the classification accuracy with each word removal, with the fastest degrade in accuracy implying the best feature saliency flagging. Hence LRP is evaluated in such a manner with the SA algorithm and with random feature removal, as showcased in detail in Chapter 4 (Section 4.3.3).

LRP and SA are both implemented for the CNN model using the iNNvestigate[16] Python framework. Moreover in case of the SVM, we follow the work conducted by Arras et al. (2017) in order to implement the LRP algorithm from first principles. Note that due to the already clear discrepancy between LRP and SA in the evaluation process for the CNN, and the constraints dictated by time limitations, we refrain from implementing the SA algorithm as well on the SVM - since this would imply the building of such a technique from first principles as well. Moreover, none of the interpretability algorithms are applied on the RF classifier. This is since during the classifier evaluation stages (which at this point were concluded), we were already aware from the experiments conducted (Section 4.2) that the RF classifier performs at an inferiority to the other two classifiers, defeating the need for further work.

Finally, a visualisation tool inspired by Arras et al. (2017) is introduced, where the classified articles are reassembled and each feature is highlighted inside of a heatmap. Red shading implies that the feature supports the hyperpartisan class, while blue shading infers that it opposes it, and white (colourless) implies that the feature is neutral in the classification. The restructuring of the original article features from the input vectors is not a trivial task to implement, and the best way we found to go about it is to separately maintain the preprocessed article tokens right before they are converted to feature representation vectors. This is done such that later on throughout the process the input vectors could be matched back to the corresponding original textual features through element-wise mapping, thereby mapping also the influence score for each feature.

[16] ***iNNvestigate*** - `github.com` [Last Accessed : 07-2020]

3.5 | System Methodology - Summary

Throughout this chapter, we described our approach for the implementation of HyperPT - automatic classification of hyperpartisan news articles. We started off in Section 3.1 by discussing a high level overview of the project - highlighting in particular the three main components forming the system. This was followed by an in-depth examination into the data sample loading and preprocessing, before the article features are transformed into representation vectors - discussed in Section 3.2. Following this, in Section 3.3 we described the implementation of three classifiers; SVM, RF and CNN for the classification of hyperpartisan news articles.

Finally in Section 3.4 we discussed the integration of external features within the classification process, namely the addition of sentiment and analysis of the classifier's decision-making, resulting in the saliency of features as an external feature.

In Chapter 4 we now evaluate the HyperPT system, in the hopes of finding the best configuration set-up for the problem at hand, while trying to achieve the best attainable performance. In doing so, we also aim at taking a closer look at the hyperpartisan article itself, with the aim of giving us further insight with which we could improve our system, and ensuing future work.

4

System Evaluation and Discussion

The dynamic arrangement discussed in Chapter 3 enables us to efficiently evaluate the HyperPT system by conducting a diversity of experiments with a level of ease and celerity.

Throughout this chapter we discuss these experiments and evaluate the components making up the HyperPT system. We start off in Section 4.1 where tests are performed to establish the best feature representation and data preprocessing techniques. This is followed by Section 4.2, where we discuss the performance of the three classification algorithms introduced to the HyperPT system; the SVM, RF and CNN, comparing them with each other and the state-of-the-art.

Having discussed the baseline classification system, we then expand in Section 4.3 on the hyperpartisan article itself, addressing the introduction of sentiment as an extra feature and the minimum length of the hyperpartisan article, along with the impact of the article title on the system performance. Moreover we evaluate the Explainable A.I. aspect of HyperPT which we then use to interpret the classification model and generate influence scores for textual features within the hyperpartisan articles. Finally in Section 4.4, the best system configuration and classification model is compared with the state-of-the-art and winner of the SemEval Hyperpartisan News Article 2019 challenge - Jiang et al. (2019).

For each of these four sections, a series of experiments are performed, followed by a succinct discussion on the results achieved and the derived implications, with the acquired knowledge then extended to the ensuing batch of experiments. In doing so we create a form of narrowing-down approach, where we maintain the best performing test configurations for the succeeding tests, and eliminate the weaker candidates. Hence, we decrease the otherwise large number of potential tests and converge towards a leading and generalised single system for the classification of hyperpartisan news articles.

Note that all tests are performed on the SemEval Hyperpartisan News Article dataset, more particularly on the By-Article collection consisting of 645 articles. A thorough background on the dataset, its origin, features and limitations is given in Section 2.1.

4.1 | Feature Representation and Data Preprocessing for Hyperpartisan News Classification

Initial tests are conducted to 1) establish the ideal feature representation with which to represent our articles and 2) determine which data preprocessing techniques would be the most effective in cleaning the article features.

We conduct experiments using a number of feature representation technologies applied alongside data preprocessing filters. We experiment with both traditional well-known feature representation methods and more modern word-embedding systems. Chosen traditional approaches include TF [Potthast et al. (2018)], TF-IDF [Papadopoulou et al. (2019)] and POS-Tags [Nguyen et al. (2019)].

Among a number of embedding systems, we retrain Word2Vec [Mikolov et al. (2013a,b)] embeddings on our dataset, while also experimenting with pretrained embeddings; pretrained Word2Vec, GloVe [Pennington et al. (2014)] and ELMo [Peters et al. (2018)]. The integration of these systems and their use inside of the HyperPT system is discussed in Chapter 3.

These listed feature representations are evaluated with five data preprocessing approaches integrated within the HyperPT system; raw text, stopword removal, punctuation removal, lowercasing and lemmatisation.

The rest of Section 4.1 is structured as follows. In Section 4.1.1 feature representation techniques are evaluated. We analyse the results acquired and choose the best performing method of all seven. We then move on to determining the most effective data preprocessing methods for the preparation of article features. This is addressed in Section 4.1.2.1, where we evaluate the performance of the five individual text cleaning approaches. Of these, the most promising ones are chosen and tested together in an aggregated manner - tackled in Section 4.1.2.2. In doing so we determine whether the integration of multiple preprocessing techniques increases or decreases the classification performance.

For any of the individual scores in the upcoming experiments, we decided on performing three separate yet identical tests, with the final result being the mean and standard deviation of the three scores. Each of these tests is performed using 10-Fold Cross-Validation, with each test value being the mean of 10 accuracy scores.

We settled on three tests since we found that it strikes the best balance between repeating the same test configurations for confidence in results and allowing us enough time to experiment with a high number of different test set-ups. We test in such rigorous fashion with the aim of evaluating the generalisation of different system configurations and increasing reliability in our assessment of the results acquired, considering the inherently stochastic nature at the core of each classification experiment.

4.1.1 | Feature Representation for Hyperpartisan News Articles

In Table 4.1, 4.2 and 4.3 we observe accuracy results obtained for each feature representation method coupled with individual preprocessing techniques. We test these couples with the aim of determining the relationship and effects of each data preprocessing and feature representation configuration. In doing so we determine the typical behaviour expected by each representation approach in use with each of the various data preprocessing methods. Based on the acquired results we then choose a number of the best performing individual preprocessing techniques, and test them simultaneously in an aggregate manner as discussed in Section 4.1.2.2.

The three tables mentioned above display feature representation and data preprocessing results respective to each of the three classifiers implemented for hyperpartisan classification; Support Vector Machine (SVM), Random Forest (RF) and Convolutional Neural Network (CNN). Since they share the same experiment configurations, we can easily map specific configuration results between any of the three classification algorithms - in doing so we not only evaluate the generalisation of every approach across the three classifiers, but also the performance of each classifier (Section 4.2).

4.1.1.1 | Feature Representation using SVM Classifier

We start by examining results obtained using the SVM classifier - showcased in Table 4.1. One instantly observes a gap in performance between ELMo embeddings and its competitors for all five preprocessing configurations. It is the only representation method to achieve accuracy scores of 0.80 or above, and does so for four of the five configurations in which it is used, with the only exception being stopword removal; with an accuracy of 0.783 and a standard deviation of ±0.009. Moreover, the highest mean accuracy of 0.813 (±0.006) for the SVM is achieved again using ELMo and removal of punctuation.

Competing word embedding technologies (corpus-trained Word2Vec, pretrained Word2Vec and GloVe) do not fare as well as ELMo, achieving results in distinguishably lower ranges and comparative to those of traditional approaches (TF, TF-IDF and POS-Tags).

Support Vector Machine

	Raw	**No Stopwords**	**No Punctuation**	**Lowercase**	**Lemmatisation**
TF	0.747 (±0.010)	0.763 (±0.008)	0.755 (±0.002)	0.751 (±0.012)	0.751 (±0.020)
TF-IDF	0.767 (±0.007)	0.758 (±0.007)	0.771 (±0.010)	0.765 (±0.019)	0.768 (±0.009)
POS-Tags	0.778 (±0.019)	0.780 (±0.006)	0.771 (±0.004)	0.787 (±0.002)	0.726 (±0.012)
Word2Vec	0.735 (±0.013)	0.744 (±0.010)	0.734 (±0.014)	0.730 (±0.010)	0.738 (±0.006)
Word2Vec (PT)	0.769 (±0.005)	0.756 (±0.007)	0.770 (±0.005)	0.762 (±0.009)	0.746 (±0.012)
GloVe (PT)	0.746 (±0.012)	0.747 (±0.014)	0.741 (±0.009)	0.737 (±0.007)	0.741 (±0.012)
ELMo (PT)	**0.800 (±0.013)**	**0.783 (±0.009)**	**0.813 (±0.006)**	**0.808 (±0.014)**	**0.800 (±0.021)**

Table 4.1: Accuracy scores of a SVM classification algorithm using individual Data Preprocessing and Data Representation configurations. Each result consists of the mean accuracy score of three identical 10-Fold Cross-Validation tests, along with the standard deviation.

Pretrained Word2Vec seem to suffer the least in accuracy, exceeding corpus-trained Word2Vec and GloVe by around 0.10 to 0.30.

POS-Tagging may perhaps be considered the most effective of the lot, exceeding all competing feature representations (with the exception of ELMo) in four out of five configurations with the sole disappointing score of 0.726 (±0.012) when used with lemmatised features (Lemmatisation). TF-IDF on the other hand seems to be a general improvement on TF, with the exception being in the case of stopword removal.

From our analysis in the case of the SVM (Table 4.1), we derive two main observations; the distinguishably better performance of ELMo to its competitors and the similar range of accuracies of all the other alternative techniques, irrelevant of their nature. In response to these two observations one must ask why is this behaviour so in the first place, and whether this is localised on the SVM or replicated for the other two classifiers; RF and CNN.

To shed some light on the first question, one must perhaps delve into the inner workings of these representation technologies. Different to Word2Vec and GloVe, ELMo feature representation vectors are contextual; wherein a feature may be represented by different dense vectors depending on the context in which it is used. Moreover, morphological clues are used for out-of-dictionary features in order to create bespoke vector embeddings [Peters et al. (2018)] (Section 2.2).

These distinctive attributes may give ELMo the extra edge in performance accuracy reflected in Table 4.1. Adding to this, competing word embedding technologies seem to fail at capturing corpus features and their contexts any better than traditional methods. Unique features inside the dataset may not be available in the vocabulary upon which the embedding technologies are trained, particularly for pretrained methods (pretrained Word2Vec, GloVe and ELMo). ELMo caters specifically for this since it uses heuristically-supported predictors to establish new, context-sensitive representation vectors.

This leaves us with the second question; whether this behaviour is replicated over other classification algorithms or if it is just manifested in the case of SVM. To investigate this, we now examine accuracy results for the RF classifier, as displayed in Table 4.2.

4.1.1.2 | Feature Representation using Random Forest Classifier

Examining accuracy results for RF in Table 4.2, one confirms that the performance monopoly of ELMo embeddings persists, albeit with a lower set of accuracy values. Moreover we notice a trend of reduced accuracy results (compared to the SVM) all throughout this batch of experiments. The highest score achieved is that of 0.785 (±0.015) using ELMo vectors and feature lemmatisation - contrasting to the highest accuracy of 0.813 (±0.006) on the SVM.

Random Forest

	Raw	No Stopwords	No Punctuation	Lowercase	Lemmatisation
TF	0.748 (±0.008)	0.761 (±0.007)	0.746 (±0.012)	0.737 (±0.008)	0.743 (±0.002)
TF-IDF	0.749 (±0.014)	0.763 (±0.007)	0.750 (±0.009)	0.750 (±0.010)	0.752 (±0.017)
POS-Tags	0.705 (±0.004)	0.694 (±0.005)	0.713 (±0.011)	0.709 (±0.001)	0.696 (±0.004)
Word2Vec	0.743 (±0.012)	0.762 (±0.002)	0.750 (±0.005)	0.750 (±0.010)	0.760 (±0.006)
Word2Vec (PT)	0.757 (±0.015)	0.732 (±0.009)	0.756 (±0.009)	0.739 (±0.004)	0.746 (±0.011)
GloVe (PT)	0.743 (±0.006)	0.734 (±0.019)	0.734 (±0.003)	0.748 (±0.009)	0.731 (±0.009)
ELMo (PT)	**0.769 (±0.012)**	**0.766 (±0.012)**	**0.771 (±0.014)**	**0.769 (±0.008)**	**0.785 (±0.015)**

Table 4.2: Accuracy scores of a RF classification algorithm using individual Data Preprocessing and Data Representation configurations. Each result consists of the mean accuracy score of three identical 10-Fold Cross-Validation tests, along with the standard deviation.

The notable performance of POS-Tags on SVM is not reciprocated in the case of RF, performing the poorest of all feature representation techniques and being the only one to break below the 0.70 accuracy mark. While the slight improvement in accuracy between TF and TF-IDF persists, so does the lack of difference between traditional and alternative embedding technologies; with both Word2Vec and GloVe scoring in the same range of results as that of TF and TF-IDF.

ELMo embeddings yet again suggest their enhanced representation capabilities over all other feature representations. The persistence of such behaviour also indicates their generalisation capabilities when used with different classification algorithms.

Having evaluated feature representation methods using both SVM and RF classifiers, we are close to converging on a single feature representation method; ELMo embeddings. Before settling on ELMo and disqualifying all other feature representation approaches, we examine accuracy results for the CNN classifier to determine whether the dominating performance of ELMo embeddings persists.

4.1.1.3 | Feature Representation using CNN Classifier

In Table 4.3, one examines classification accuracies for feature representation approaches using the CNN classifier.

Here we see a variation in the range of results not previously seen in previous experiments, with accuracy scores ranging from as low as 0.567 (±0.008) to as high as 0.790 (±0.018). The pattern noticed for ELMo embeddings in preceding experiments is repeated here as well, achieving the highest accuracy results for nearly all data preprocessing configurations except for lemmatisation - where ELMo is exceeded in performance by corpus-trained Word2Vec.

Convolutional Neural Network

	Raw	**No Stopwords**	**No Punctuation**	**Lowercase**	**Lemmatisation**
TF	0.623 (±0.022)	0.605 (±0.008)	0.634 (±0.005)	0.636 (±0.011)	0.642 (±0.008)
TF-IDF	0.648 (±0.005)	0.602 (±0.016)	0.647 (±0.009)	0.651 (±0.008)	0.644 (±0.014)
POS-Tags	0.573 (±0.007)	0.567 (±0.008)	0.589 (±0.008)	0.568 (±0.006)	0.668 (±0.004)
Word2Vec	0.761 (±0.007)	0.772 (±0.005)	0.765 (±0.007)	0.766 (±0.005)	**0.771 (±0.006)**
Word2Vec (PT)	0.746 (±0.011)	0.755 (±0.019)	0.759 (±0.018)	0.758 (±0.001)	0.742 (±0.002)
GloVe (PT)	0.773 (±0.017)	0.759 (±0.010)	0.749 (±0.009)	0.770 (±0.012)	0.762 (±0.014)
ELMo (PT)	**0.790 (±0.018)**	**0.773 (±0.010)**	**0.778 (±0.012)**	**0.787 (±0.008)**	0.766 (±0.004)

Table 4.3: Accuracy scores of a CNN classification algorithm using individual Data Preprocessing and Data Representation configurations. Each result consists of the mean accuracy score of three identical 10-Fold Cross-Validation tests, along with the standard deviation.

Moreover corpus-trained Word2Vec, for the first time, distinguishes itself from the other embedding representations (pretrained Word2Vec and GloVe). For the first time as well, we see a difference between embedding and traditional representation technologies, due to the fact that traditional methods achieve very poor results, with the worst performer being yet again POS-Tags.

Indeed none of the traditional methods achieve results at or over the 0.70 accuracy mark, which is a far cry from the notably better results achieved in previous experiments. POS-Tags are without question the most controversial of the lot with the highest scores among traditional approaches for SVM and the lowest for both RF and CNN.

Comparing these experiments to their predecessors, the only variable to change is the classifier itself. Being the only Deep Learning (DL) classifier of the three, one wonders whether the DL nature of CNN plays a part in this outcome. This along with the performance of the SVM and the RF, is discussed further in Section 4.2.

In Section 4.1.1.1, Section 4.1.1.2 and Section 4.1.1.3, we discussed experiments conducted on the three classification algorithms. Now we address the overall performance of tested feature representation techniques to reach a conclusion as to the best approach for hyperpartisan news classification.

4.1.1.4 | Feature Representation for Hyperpartisan News Articles - Overview

From our analysis of accuracy scores in Table 4.1, Table 4.2 and Table 4.3, we observe ELMo embeddings distinguishing themselves as the best tested feature representation approach. They achieve the highest accuracy in all individual data preprocessing configurations tested, with standard deviation values maintained in the same range as that of competing data representation systems.

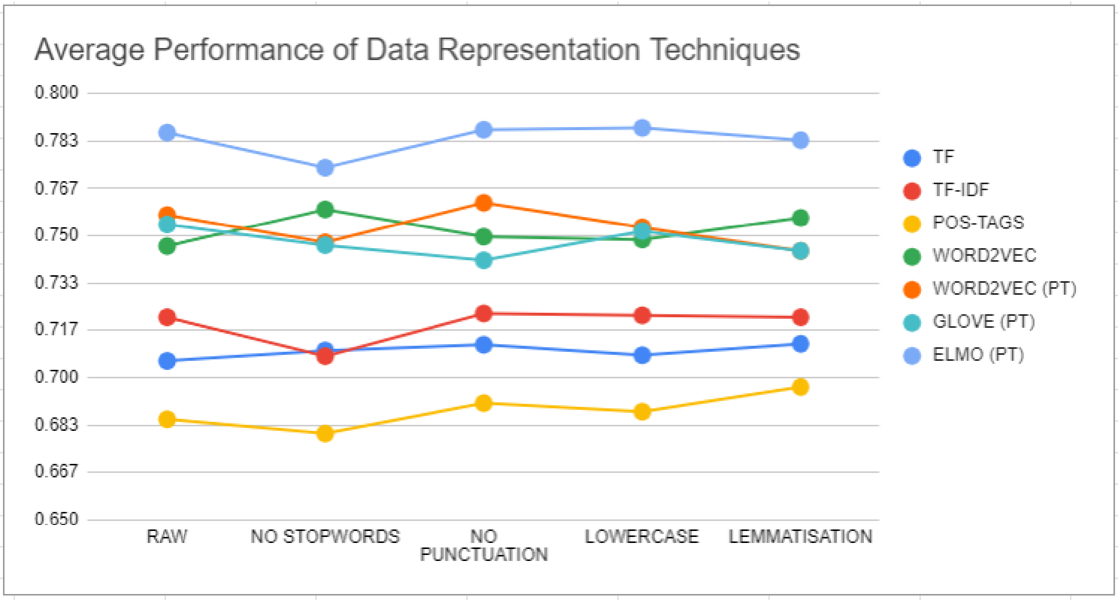


Figure 4.1: Accuracy results for individual Data Representation techniques averaged over the three classification algorithms; SVM, RF and CNN.

The superior performance of ELMo can perhaps be better visualised in Figure 4.1, where we plot the mean accuracy scores of tested data representation techniques as a time-series chart. The pretrained ELMo embeddings are completely segregated at the

0.783 level, while competing word-embedding technologies are altogether clustered at the 0.750/0.760 ranges, achieving more or less the same range of results.

Observing Figure 4.1, all tested word-embedding technologies lie at a higher range to traditional data representation systems - in turn lying in the lower ranges of 0.680 to 0.723. As expected from our observations in Section 4.1.1.3, POS-Tags result in the worst performers, undoubtedly affected by their notably poor performance when tested using the CNN classifier (Table 4.3). Moreover, all traditional data representation techniques suffer when used in conjunction with the CNN, resulting in a heavy impact on their averaged performance.

We hence conclude that pretrained ELMo Embeddings are the most effective feature representation approach for all three classifiers. In doing so we discontinue experiments on all other data representation methods and establish ELMo as the chosen data representation technology for our system.

4.1.2 | Data Preprocessing for Hyperpartisan News Articles

In Section 4.1.1 we performed experiments on various feature representation techniques - converging on ELMo embeddings [Peters et al. (2018)] as the best performing feature representation approach out of the seven tested for the classification of hyperpartisan news articles.

Tested representations are simultaneously tested with five individual data preprocessing/cleaning methods; 1) raw textual features (no preprocessing applied), 2) removal of stopwords, 3) removal of punctuation, 4) lowercasing of features and 5) lemmatisation of features. In the upcoming Section 4.1.2.1, we discuss the performance of each of these approaches with respect to the three classifiers. We re-examine the accuracy scores achieved using ELMo embeddings with each of the aforementioned preprocessing approaches - in doing so evaluating their generalised performance.

From these results we choose a number of the best performing preprocessing techniques, to then test them together in Section 4.1.2.2 as an aggregate preprocessing function applied on the corpus features. This step is applied since unlike feature representation approaches, data preprocessing methods can be, and often are, applied simultaneously on the same features as an aggregate preprocessing procedure.

4.1.2.1 | Individual Data Preprocessing for Hyperpartisan News Articles

In order to examine the performance of individual data preprocessing techniques, we extract accuracy results from Table 4.1, 4.2 and 4.3 corresponding to each of the tested data preprocessing approaches in conjunction with ELMo embedded features. In doing so we examine preprocessing methods at their proven best performance - when used with ELMo feature representations (Section 4.1.1).

In Table 4.4 one can observe these scores, along with the mean accuracy and standard deviation of each technique across all three classification algorithms; the SVM, RF and CNN.

	Raw	**No Stopwords**	**No Punctuation**	**Lowercase**	**Lemmatisation**
SVM	0.800 (±0.013)	0.783 (±0.009)	0.813 (±0.006)	0.808 (±0.014)	0.800 (±0.021)
RF	0.769 (±0.012)	0.766 (±0.012)	0.771 (±0.014)	0.769 (±0.008)	0.785 (±0.015)
CNN	0.790 (±0.018)	0.773 (±0.010)	0.778 (±0.012)	0.787 (±0.008)	0.766 (±0.004)
Mean	0.786 (±0.014)	0.774 (±0.010)	0.787 (±0.011)	**0.788 (±0.010)**	0.784 (±0.013)

Table 4.4: Individual and mean accuracy scores for all tested data preprocessing techniques using ELMo embeddings. The mean results consist of the mean performance of each data preprocessing approach when used in conjunction with ELMo embeddings across the 3 classifiers.

As is also observed in Section 4.1.1, we notice that the best results are achieved using SVM, with four out of five preprocessing methods scoring at or above the 0.80 mark. The highest accuracy for both the SVM as well as the rest of the experiments is that of 0.813 (±0.006) - acquired using non-punctuated features.

Between each set of per-classifier results we notice a reoccurring pattern where the highest score is typically achieved using SVM, with the lowest being with the RF, and the CNN lying in between. This pattern holds true for four out of the five preprocessing approaches, with lemmatisation being the exception where the RF classifier score of 0.785 (±0.015) exceeds that of the CNN's 0.766 (±0.004). This trend suggests that the SVM is perhaps the most adequate for handling the classification problem at hand, with the RF being the poorest. The same observation is also noted in experiments on feature representation approaches in Section 4.1.1, and is discussed further in Section 4.2.

We evaluate the overall performance of preprocessing techniques by examining the mean accuracy for each across all three classifiers as shown in Table 4.4. Albeit scoring the highest individual score, No Punctuation settles at second place, with lowercasing of features getting the slightly higher mean accuracy of 0.788 (±0.010) to No Punctuation's 0.787 (±0.011). The reason behind this lies in the results acquired using CNN, where Lowercase achieves significantly higher accuracy to No Punctuation, balancing out the otherwise slightly poorer performance in the case of both SVM and RF.

Raw and lemmatised textual features follow closely behind at 0.786 (±0.014) and 0.784 (±0.013), with the worst performer being stopword removal at 0.774 (±0.010) -

being also the only preprocessing method to score an accuracy below 0.80 (at 0.783) on the SVM classifier.

Due to No Punctuation and Lowercase achieving the highest two mean accuracies, one could argue that reducing the ’clutter’ and cleaning the corpus text such that it consists of simple lowercased textual features facilitates the classification process; further paving the way for the classifier and thereby increasing the system’s performance.

One must be careful however not to remove too many corpus features such that it impairs the classification performance. We believe this is the case for No Stopwords. The removal of stopwords may very well be decreasing the inherent perception of hyperpartisan articles, since albeit stopwords seeming trivial at first glance, the context in which a stopword is used may significantly affect the meaning the article is trying to portray.

From our observations above, we choose No Punctuation, Lowercase, Raw textual features and Lemmatisation as the most promising individual contestants. We thereby opt to leave out solely the worst performer; No Stopwords, from aggregate preprocessing testing. In Section 4.1.2.2 we conduct testing using aggregate preprocessing techniques and apply them simultaneously on the corpus features. In doing so we determine whether aggregate solutions perform better or worse than individual ones for the detection of hyperpartisan news articles.

4.1.2.2 | Aggregate Data Preprocessing for Hyperpartisan News Articles

Further to the developments on data preprocessing in Section 4.1.2.1, we now combine the four chosen approaches in a number of aggregate preprocessing operations, and compare accuracy performances with that of individual preprocessing methods. In doing so we test out whether aggregate cleaning of corpus features fares better or worse than simpler, individual filtering - before eventually converging on a single, best approach.

In Table 4.5 we observe classification results using both individual and aggregate feature preprocessing, where we finally compute the mean accuracy across all three classifiers. Individual preprocessing approaches consist of the four inherited from our experiments in Section 4.1.2.1; Raw text, No Punctuation (NP), Lowercase (LC) and Lemmatisation (LM). Aggregate preprocessing solutions are then combinations of the latter three; No Punctuation + Lowercase (NP + LC), No Punctuation + Lemmatisation (NP + LM), Lowercase + Lemmatisation (LC + LM), and finally No Punctuation + Lowercase + Lemmatisation (NP + LC + LM).

One immediately notices the inferior performance of aggregate solutions to individ-

Accuracy Results for Aggregate Preprocessing Techniques

	SVM	**RF**	**CNN**	**Mean**
Raw	0.800 (±0.013)	0.769 (±0.012)	0.790 (±0.018)	0.786 (±0.014)
No Punctuation (NP)	0.813 (±0.006)	0.771 (±0.014)	0.778 (±0.012)	0.787 (±0.011)
Lowercase (LC)	0.808 (±0.014)	0.769 (±0.008)	0.787 (±0.008)	**0.788 (±0.010)**
Lemmatisation (LM)	0.800 (±0.021)	0.785 (±0.015)	0.766 (±0.004)	0.784 (±0.013)
NP + LC	0.788 (±0.015)	0.765 (±0.002)	0.763 (±0.006)	0.772 (±0.008)
NP + LM	0.777 (±0.006)	0.773 (±0.006)	0.754 (±0.014)	0.768 (±0.009)
LC + LM	0.791 (±0.021)	0.778 (±0.009)	0.774 (±0.012)	0.781 (±0.014)
NP + LC + LM	0.794 (±0.016)	0.763 (±0.017)	0.772 (±0.019)	0.776 (±0.017)

Table 4.5: Accuracy results and their standard deviations for individual and aggregate Data Preprocessing techniques using ELMo Embeddings. All scores are the mean of three separate and identical tests, with each being the score of a 10-Fold Cross-Validation.

ual ones; with solely (NP + LM) and (LC + LM) scoring higher than some individual approaches and only when applied along with the RF classifier. Conversely, the rest of the aggregate results are poorer than their counterparts, especially in the case of the SVM and CNN.

The best performing aggregate solution out of all would be (LC + LM) - achieving the highest score of the lot with all three classification algorithms. This is further elucidated when examining the mean accuracies over the three classifiers, where (LC + LM) is the only aggregate solution to achieve a mean score higher than 0.780 [at 0.781 (±0.014)]. This score still falls short however, compared to any of the individual approaches, with the highest mean score of all experiments in Table 4.5 being Lowercase at 0.788 (±0.010) and No Punctuation following close behind at 0.787 (±0.011).

Given these observations we eliminate all aggregate solutions and focus on individual ones. The decision for the ultimate data preprocessing procedure lies between No Punctuation and Lowercase, with the two having respective mean accuracies of 0.788 and 0.787. Examining the mean standard deviations of each, we find a slightly higher deviation for No Punctuation (at 0.011) than for Lowercase (at 0.010). Albeit the difference being such a small margin, this does offer the suggestion that Lowercase also tends to vary less (albeit not by much) between experiments, making it more reliable.

Keeping this in mind we also notice a slightly higher performance in aggregate preprocessing solutions containing the lowercasing of features. Examining the mean accuracies of the four aggregate approaches, one notices that the lowest score of the lot (0.768) is achieved with (NP + LM) - being the only procedure not containing lowercased features. Moreover, adding Lowercase to this combination such that we have (NP

+ LC + LM) boosts the mean accuracy by nearly 10% to an accuracy of 0.776 - further validating the positive effects of feature lowercasing.

The notable effectiveness of feature-lowercasing and the reasons behind it are interesting points to investigate. Since lowercasing tends to reduce the size of the corpus vocabulary due to some uppercased and capitalised features being the same as other lowercased ones, we think this may perhaps decrease the complexity and variety of the corpus articles, particularly for such a small dataset as the SemEval Hyperpartisan News Articles dataset (Section 2.1). The reduction in complexity may increase the classification accuracy due to the article being simpler for the classification algorithm to work with.

From the results acquired we hence conclude on lowercasing of all corpus features as the ideal preprocessing stage before hyperpartisan news classification is performed. In doing so we incorporate this procedure in all of the ensuing tests and eliminate all the other contestants.

4.1.2.3 | Data Preprocessing for Hyperpartisan News Articles - Overview

Throughout Section 4.1.2 we discussed and evaluated a variety of data preprocessing techniques for the classification of hyperpartisan news articles.

We started off in Section 4.1.2.1 with individual, simplistic data cleaning approaches, namely; raw untouched features, removal of stopwords, removal of punctuation, lowercasing and finally lemmatisation of corpus features. We performed classification experiments with each of three classifiers - SVM, RF and CNN. In doing so we evaluated the classification performance of each, to finally eliminate the removal of stopwords due to its inferior performance (refer to Table 4.4) before proceeding to experiment with aggregate preprocessing techniques in Section 4.1.2.2.

Aggregate solutions consist of different combinations of multiple individual preprocessing approaches applied simultaneously on article features. Four aggregate solutions are tested in all and compared with each other as well as with their individual counterparts (refer to Table 4.5). Despite the added complexity they fail to outperform accuracies achieved using simpler individual techniques at almost every scenario.

Observing both individual and aggregate performances, we finally settle on lowercasing of article features as the ideal data preparation process before the article features are fed to the corresponding classification algorithms.

4.1.3 | Feature Representation and Data Preprocessing for Hyperpartisan Classification - Discussion

In Section 4.1 we evaluated ideal feature representation and data preprocessing approaches for data preparation before classification could be performed.

Initial experiments in Section 4.1.1 consisted of establishing one feature representation with which to efficiently represent hyperpartisan article features. Both traditional and word-embedding technologies were tested, with traditional features consisting of TF, TF-IDF and POS-Tags, and word-embedding approaches namely corpus-trained and pretrained Word2Vec, pretrained GloVe and finally pretrained ELMo.

ELMo was established as the best performing feature representation approach for our articles features, with a clear improvement on performance compared to any of the other approaches (refer to Figure 4.1). Hence ELMo is maintained throughout all ensuing tests as the go-to feature representation of hyperpartisan news article features.

Having established an ideal feature representation approach, we then focused on repeating the same sets of experiments to determine ideal data preprocessing techniques (if any) for the preparation of article features. This was tackled in Section 4.1.2, where we discussed the performance of both individual and aggregate preprocessing solutions.

Analysis of acquired accuracy results implies that removal of punctuation and lowercasing of features come head to head as the two best preprocessing techniques, with lowercasing of features performing slightly better and being more consistent in nearly all experiments, thereby establishing itself as the go-to data preprocessing application for our hyperpartisan news classification system.

In Section 4.1 we established the ideal feature representation and data preprocessing approaches for the classification of hyperpartisan news articles. In doing so we conclude the initial stages of our experimentations, extending the knowledge acquired and the established methods to the succeeding segment; Section 4.2, where we review the already conducted experiments and moreover perform new tests to evaluate the performance of classification algorithms with the hopes of converging on the best performing one to be used throughout the HyperPT system.

4.2 | Classifier for Hyperpartisan News Classification

In our previous discussions we established ELMo embeddings [Peters et al. (2018)] as the most effective approach for representing our corpus features, which, as confirmed also in Section 4.1, should be lowercased so that their maximum potential is utilised.

We now examine the classification algorithms themselves, namely the SVM, RF and CNN. Throughout this section we analyse the nature of these three algorithms, compare their performance with that of one another, and highlight the strengths and weaknesses of each to finally decide on a single approach.

The rest of this section is structured as follows. In Section 4.2.1 we apply hyperparameter tuning on each of the three algorithms listed above with the aim of establishing the best hyperparameter configuration for each. Afterwards in Section 4.2.2 we proceed to evaluate the classification performance of each classifier, in doing so re-examining experiments conducted in Section 4.1 along with further new testing. Finally we remark our conclusions and verdict on the ideal classification algorithm for hyperpartisan news articles - detailed in Section 4.2.3.

4.2.1 | Hyperparameter Tuning for Hyperpartisan News Classification

We perform hyperparameter tuning on each classification algorithm (SVM, RF and CNN) with the aim of establishing the ideal configuration for maximising the classifier performance. Note that in practice hyperparameter tuning is conducted in conjunction with initial experimentations on different feature representations and data preprocessing techniques; such that the classifier is already optimised and tweaked for any state of the preprocessed features.

Some minor changes to the classifier hyperparameters are recommended depending on the applied feature representations and data preprocessing, however 1) The major hyperparameter configurations are clearly established no matter which preprocessing configuration is used and 2) In order to maintain our focus on the established preprocessing approaches (Section 4.1), we focus on hyperparameter tuning with respect to lowercased features represented as ELMo embedded vectors.

In conducting our experiments we implement a Grid Search optimisation approach, where a *grid* of different hyperparameter possibilities is generated, following which all combinations are individually tested. Random Search was also considered at this stage, however considering that 1) the number of hyperparameters is not very large and 2) the testing itself is not too time consuming; we decided on testing out all possible configurations and leave no room for doubt.

We test using 10-Fold Cross Validation, similar to all preceding testing discussed in Section 4.1. In doing so we evaluate the reliability of each hyperparameter configuration and its generalisation abilities while maintaining a high level of performance over different hyperpartisan news articles. SKLearn's GridSearchCV[1] tool is utilised to automate the above process; exhaustively searching for the ideal hyperparameter configuration using 10-Fold Cross Validation testing.

4.2.1.1 | Hyperparameter Tuning the SVM Classifier

The SVM classifier is tested on three separate hyperparameters; 1) the SVM kernel, 2) the kernel coefficient γ and 3) the regularisation parameter λ. We perform our experiments on the four main types of SVM kernels; Linear, Sigmoid, Polynomial and RBF. All of these kernels with the exception of Linear (since it does not require a kernel coefficient) are evaluated using four γ samples as displayed in Table 4.6.

The four samples for γ and the four for λ are generated from a log scale (10^x), where x is respectively between -5 to 3, and -2 to 3. This logarithmic distribution allows us to explore the range of possible values quicker by scaling with an equal scale of difference with each candidate value [Joshi et al. (2016); Pedregosa et al. (2011)]. More on the SVM and the corresponding hyperparameters can be perused in Chapter 2.

Hyperparameter	**Samples**
Kernel	Linear Sigmoid Polynomial RBF
Kernel Coefficient γ	1.00×10^{-5} 4.64×10^{-3} $2.15 \times 10^{+0}$ $1.00 \times 10^{+3}$
Regularisation Parameter λ	1.00×10^{-2} 4.64×10^{-1} $2.15 \times 10^{+1}$ $1.00 \times 10^{+3}$

Table 4.6: SVM classifier hyperparameters evaluated using Grid-Search hyperparameter tuning.

Initial testing including a variety of preprocessing approaches (Section 4.1) suggested a close tie between the Linear kernel and the more complex RBF kernel. The other two kernels (Sigmoid and Polynomial) are very rarely chosen through Grid-Search, with multiple experiments converging on either the Linear or the RBF kernels.

Shifting our focus to ELMo embeddings with lowercased vectors we see a different outcome, where the RBF kernel dominates over multiple experiments as the most

[1] *SKLearn GridSearchCV* - `www.scikit-learn.org` [Last Accessed: 06-2020]

adequate SVM kernel for the classification of hyperpartisan news articles. Moreover a kernel coefficient γ of 4.64×10^{-3} is suggested as the ideal accompanying coefficient, with the SVM regularisation parameter λ at $2.15 \times 10^{+1}$. Over the substantial number of tuning experiments conducted on the SVM, this configuration achieves a mean accuracy of 0.805 and a highest accuracy of 0.840.

4.2.1.2 | Hyperparameter Tuning the RF Classifier

Compared to the SVM, the RF classifier offers us a wider range of hyperparameters to tune. Moreover the seemingly poorer overall performance of the algorithm discussed back in Section 4.1 further compels us to perform extensive hyperparameter tuning. In Table 4.7 we observe the hyperparameters tuned for the RF classifier.

Hyperparameter	**Samples**
Number of Estimators	50 200 500
Maximum Tree Depth	*3* *5* *None*
Maximum Features	1 10 *Auto*
Minimum Sample Split	2 3 10
Bootstrapping	*True* *False*
Criterion	Gini Entropy

Table 4.7: RF classifier hyperparameters evaluated using Grid-Search hyperparameter tuning.

We again test out all hyperparameter combinations using Grid-Search with 10-Fold Cross Validation. RF hyperparameters include:

- The number of estimators/trees to use during classification.
- The maximum tree depth, which when set to *None* implies that no limit is set and is expanded as much as necessary.

- The maximum amount of features to consider when looking for the best split. If set to *Auto*, $\sqrt{num.features}$ is considered by default.
- The minimum number of samples necessary to split a tree node.
- Whether to bootstrap the classifier or not. If so, samples are used to build the classifier trees - if not, the whole dataset is used.
- The criterion with which to measure the quality of each node split.

Grid-Search results acquired from multiple experiments indicate that the Entropy criterion performs best in nearly all cases, particularly for ELMo embedded and lowercased features. A high numbers of estimators is suggested, with test Grid-Search results varying between 200 and 500. Despite an estimator count of 200 obtaining less erratic results, the highest accuracy of 0.796 is achieved with 500 estimators, with 200 estimators following closely behind at 0.793.

In all experiments conducted, the maximum tree depth is strongly suggested to be kept to *None* (thereby being expanded as much as necessary). The maximum number of features is also to be maintained at *Auto*, implying again a dynamic adaptation to the number of features during runtime. Interestingly, a minimum samples split of 2 is maintained for tests recommending 200 estimators, while a samples split of 10 is emphasized for 500 estimators. Bootstrapping is not recommended in any of the tests, implying the use of the whole dataset for the generation of each tree.

4.2.1.3 | Hyperparameter Tuning the CNN Classifier

The CNN implementation is based on the state-of-the-art introduced by Jiang et al. (2019); consisting of five convolutional layers followed by Batch Normalization and a dense classification layer (Chapter 2). Despite this static structure, we maintain a number of experiments on other aspects of the network with the hopes of enhancing the algorithm's performance and tune it further to the problem at hand.

As observable in Table 4.8, four hyperparameters are tested; the ideal number of training epochs, the batch size at which data batches are inserted, whether to use batch normalisation (as suggested by Jiang et al. (2019)), and whether to use early stopping (with differing epoch tolerances). Much the same as the tuning for the other two approaches, Grid-Search using 10-Fold Cross Validation is used for the monitoring of results and performance.

From the acquired results we notice a few similarities with the system proposed by Jiang et al. First off the number of epochs is maintained at the relatively low value of 30,

Hyperparameter	**Samples**
Number of Epochs	30 50 100
Batch Size	15 32 50
Batch Normalisation	*True* *False*
Early Stopping	Tolerance of 2 Epochs Tolerance of 5 Epochs *None*

Table 4.8: CNN classifier hyperparameters evaluated using Grid-Search hyperparameter tuning.

suggesting a possibility of overfitting with more epochs. The same can be said for the batch size, which is kept the same at 32. Batch Normalisation is the third parameter to be preserved - being recommended in every test iteration.

Finally we get to early stopping. It seems that in our case such a regularisation technique is more of a hindrance than an aid, leading us to note that with both tolerances of 2 and 5 epochs the algorithm training is perhaps still terminated prematurely. We believe that particularly in the case of the CNN, the size of the dataset upon which it is trained does have a strong affect not only the classifier's performance but also the hyperparameter preference. We think that the particularly negative outcome experienced throughout tests involving early stopping to those consisting of full training is a direct result of the small nature of our dataset (SemEval Hyperpartisan news articles), implying a tendency for the classifier to be under-fitted on the data.

Having said so, the low number of training epochs should, in our case, keep the possibility of over-fitting at bay. A mean accuracy of 0.788 and a highest accuracy of 0.830 is achieved during hyperparameter tuning using the proposed hyperparameter configuration.

4.2.1.4 | Hyperparameter Tuning - Discussion

In Section 4.2.1 we determined the ideal hyperparameter configuration for each of three classifiers; SVM, RF and CNN. The conclusions drawn are based on lowercased article features represented as ELMo embedded vectors - as chosen by initial experiments conducted in Section 4.1.

In tuning the hyperparameters for the SVM, we determined that RBF is the best performing kernel. The use of this kernel is coupled with a kernel coefficient γ of 4.64×10^{-3} and a misclassification regularisation parameter λ of $2.15 \times 10^{+1}$.

In the case of RF, Entropy is preferred to Gini as the measure of impurity at each node. The lack of bootstrapping slows down the training time yet increases performance accuracy. Finally among other hyperparameters, we decide on moving forward with 200 estimators (trees), since from the experiments conducted we find that it is the most generalised approach.

Finally we evaluated the CNN hyperparameters to discern that the majority of the resulting parameter configurations mirror those of the state-of-the-art presented by Jiang et al. (2019). This holds true for the number of training epochs - set at 30, the batch size, set at 32, and the confirmed effectiveness of batch normalisation within the classifier. Reaching the same hyperparameter configurations as the state-of-the-art gives us trust in our work, since it indicates that our implementation is of the quality and precision necessary to reach the same conclusions as the state-of-the-art.

4.2.2 | Evaluating HyperPT Classifier performance

Following hyperparameter tuning for our classification algorithms, in Section 4.2.2 we evaluate the effectiveness of these classifiers.

In Section 4.2.2.1 we first discuss the performance of the three tested classifiers with that of one another. Using these observations we then reach our conclusion on the most adequate classification algorithm for the classification of hyperpartisan news articles.

The now decreased number of tests from the initial preprocessing experiments conducted in Section 4.1 make it feasible to also take into consideration the F1 score in conjunction with the accuracy. F1 (Section 2.6) is a measure which gives us the weighted mean between the precision and recall. Such a statistic is important in evaluating the performance of different classifiers, since it informs us how generalised the classifier is in its predictions.

4.2.2.1 | Evaluating Classifier performance on Hyperpartisan News Classification

We display the corresponding accuracy results for our three classifiers in Table 4.9. Here we list both the mean and highest, accuracy and F1 scores for each classification algorithm, be it; the SVM, RF, or CNN. Moreover, we list as well the highest score for each in case of comparisons with other research which may include the highest scores achieved.

We examine two sets of experiments for each classifier; one using lowercased features (+ LC), since lowercasing of features was chosen (back in Section 4.1.2) as the best performing general data preprocessing, and another using alternative preprocessing based on the ideal classifier performance (observed in Section 4.1). Here we find no punctuation (+ NP), lemmatisation (+ LM) and finally raw features (+ Raw).

Classifier	Mean Accuracy	Mean F1 score	Highest Accuracy	Highest F1 score
SVM + LC	0.808 (±0.013)	0.722 (±0.009)	0.809	**0.730**
SVM + NP	**0.813 (±0.006)**	**0.722 (±0.006)**	0.821	0.726
RF + LC	0.769 (±0.008)	0.648 (±0.019)	0.777	0.659
RF + LM	0.785 (±0.015)	0.646 (±0.005)	0.795	0.650
CNN + LC	0.787 (±0.008)	0.699 (±0.019)	0.797	0.713
CNN + Raw	0.790 (±0.018)	0.708 (±0.013)	**0.822**	0.719

Table 4.9: Mean and Highest accuracy and F1 results for each classification algorithm. Two sets of tests are performed for each classifier; one using lowercased features and the other using data preprocessing approaches with which initial scores were the highest. All features are represented as ELMo Embeddings.

We observe in Table 4.9 that the best mean accuracy, that of 0.813 (±0.006), is maintained by the SVM using No Punctuation (NP), with the highest accuracy achieved being that of 0.821. These scores are accompanied by the mean and highest respective F1 scores of 0.722 (±0.006) and 0.726. One notices that the SVM seems to be more adapted to non-punctuated than lowercased features, achieving better accuracies at just about the same F1 scores.

Despite a significant reduction in mean accuracy, CNN classification on raw features achieves the highest accuracy score throughout - that of 0.822. Compared with the results discussed above, the difference in accuracy (0.001) may be considered negligibly small, more so since 1) a lower F1 score implies that the CNN is not as balanced as the alternative and 2) higher deviations from the mean accuracy and F1 imply a decrease in the consistency of the CNN performance.

Training on such a small number of data points as our dataset (Section 2.1) may very well be limiting the CNN from reaching its full potential - resulting in similar or poorer accuracy ranges to those of alternative traditional approaches. This phenomenon is also documented in published literature [Arras et al. (2017); Zhang et al. (2015)], where the authors note that the performance of the CNN truly becomes superior to that of TFIDF-based traditional systems when trained on large datasets typically consisting of millions of data samples.

From our discussion above we find that the SVM would be the better choice for classifying hyperpartisan news articles, when compared with both the CNN and RF. We base our conclusion on the SemEval Hyperpartisan dataset upon which this study is conducted, however we do not take into consideration solely the performance metrics put forward in Table 4.9, but also the flexibility, resource requirements and feasibility of each method.

From the accuracy results analysed above, it is clear that the SVM consistently reaches higher mean accuracies than the CNN and the RF. It fits the classification problem better, as judged by higher all-round F1 scores. Moreover standard deviations included with the mean metric values suggest less variations in accuracy results over multiple experiments.

Despite their superior performance, DL classifiers take comparatively longer to train and require both larger amounts of training data and more advanced hardware configuration. These prerequisites are not justified by the above results, making the SVM a more favourable approach not only due to the better classification performance but also due to simply requiring less resources, less time and less data. Hence we conclude on the SVM as our classifier for the detection of hyperpartisan news articles.

Despite establishing lowercasing of features as the recommended overall preprocessing step over all tested classifiers (Section 4.1.2), one notices that in the specific case of the SVM, removal of punctuation seems to work better - typically resulting in a higher and more balanced classification performance. Thereby we conclude that if one is set on using the SVM as the classification algorithm, such as in our case, non-punctuated features may very well be a better option.

We observe that the reason behind this increased performance may be since the removal of punctuation decreases the ambiguity and complexity from the textual sequence, which for classifiers (such as the SVM) not taking context into consideration, facilitates the classification process. In ensuing tests, we hence consider the removal of punctuation from article features as much as lowercasing.

4.2.3 | Classifier for Hyperpartisan News Classification - Discussion

In Section 4.2 we discussed the performance of the three classification algorithms (SVM, RF and CNN) employed within the HyperPT system for the classification of hyperpartisan news articles. Having tuned our classifier hyperparameters as specified in Section 4.2.1, we then observed and compared the accuracy results and F1 scores (Section 4.2.2).

From these outcomes we concluded on the SVM as the best performing classification algorithm for the classification of hyperpartisan news articles. This conclusion is supported by the proven superior performance of the SVM - having the highest mean accuracy (0.813 [±0.006]) and highest mean F1 score (0.722 [±0.006]). The RF seems to struggle to properly fit the problem at hand, presenting the poorest performance of the three. Moreover, the simpler, more feasible nature of the SVM compared to the CNN makes it more feasible to handle and maintain.

With the evaluation of the classifiers we conclude the evaluation of the baseline classification system employed within HyperPT. In Section 4.3.1 we explore external features with the aim of embedding them within the classification process to investigate their effect on the system performance and in doing so delve further into the typical nature of a hyperpartisan article.

4.3 | Effectiveness of Sentiment and additional Features on Hyperpartisan News Classification

In Section 4.1 we established ELMo embeddings and lowercasing of data features as the best performing feature representation and generalised preprocessing approaches across the three tested classifiers for the classification of hyperpartisan news articles. In the succeeding Section 4.2 we tuned our classifiers and settled on the best performing one; the SVM.

Throughout this section we experiment with a number of non-conventional features, with the aim of investigating the resultant effects on the overall performance of the classification, and in doing so exploring further the typical nature of such news articles.

In Section 4.3.1 we first discuss the addition of sentiment within the article textual features. We experiment with sentiment embedding techniques to investigate the varieties in classification performance. Finally we conclude on whether, from the results acquired, to maintain sentiment features within the HyperPT system, or discard them.

Following the evaluation of sentiment, we investigate (Section 4.3.2) the effects of differing article lengths, and the addition of the article title. This is followed by Section 4.3.3, where we measure the degree of saliency (influence) of each feature within by interpreting the corresponding classification. In doing so we not only observe the typical features supporting or opposing hyperpartisanship, but also observe the classifiers' behaviour in generating the classification labels.

4.3.1 | Sentiment as a Feature for Hyperpartisan News Classification

Agendas behind hyperpartisan news articles often aim at sensationalising ideas about the topics being presented (Chapter 1). We hence believe it is crucial to investigate the integration of sentiment represented as external features embedded within corpus articles, with the aim of determining whether sentiment indeed plays a notable role in classifying hyperpartisan news.

We perform tests on thirteen different sentiment feature configurations, with the aim of benchmarking the approaches with the best non-sentiment classification results and possibly establish the best sentiment feature composition (if any). In doing so, we also evaluate the system's performance, balance and reliability with the integration of sentiment.

As discussed in Section 3.4.1, we experiment with sentiment as a compounded score, a negative score, and a textual label (positive/negative) at the article level and at the sentence level. We include negative scores by themselves along with the compound score since other studies [Anthonio and Kloppenburg (2019)] report increased classification performance using solely negative sentiment scores, rather than positive or compounded.

Furthermore, we perform additional experiments where sentiment scores are scaled by a factor of 1000 in order to determine whether such scaling of sentiment weights affects the classification performance, and in what way. The value of 1000 was chosen since we felt that this strikes a balance between proper scaling of the sentiment scores, without over-exaggerating the score values. Lastly we conduct tests with all VADER scores (positive, negative, neutral and compound) embedded within each article, a technique inspired by Palić et al. (2019).

Following the conclusive results obtained in Section 4.1 and Section 4.2, we perform our sentiment analysis experiments using ELMo embeddings along with the SVM classifier. Separate yet identical sets of tests are performed on the corpus stripped of any punctuation and the corpus with lowercased features, since, as discussed earlier, these two preprocessing approaches have been proven to perform the best with this classifier. Corresponding accuracy and F1 scores can be examined in Table 4.10.

4.3.1.1 | Evaluating Sentiment as a Feature for Hyperpartisan News Classification

Examining performance results in Table 4.10, one notices that in none of the tests involving sentiment is the corresponding accuracy or F1 score comparative to or exceeding scores achieved without sentiment. This holds true for both lowercased and nonpunctuated features. Indeed the main characteristics of tests involving sentiment are a reduction in accuracy and F1 score, with higher standard deviations from the mean scores. This would imply that the SVM struggles more to fit the classification problem

Lowercase		
Sentiment Features	**Accuracy**	**F1 Score**
No Sentiment	**0.808 (±0.013)**	**0.722 (±0.009)**
Global Label - Article	0.757 (±0.008)	0.617 (±0.013)
Derived Label - Article	0.732 (±0.085)	0.637 (±0.035)
Compound Score - Article	0.773 (±0.014)	0.643 (±0.019)
Compound Score ×1000 - Article	0.778 (±0.013)	0.655 (±0.025)
Negative Score - Article	0.781 (±0.003)	0.660 (±0.006)
Negative Score ×1000 - Article	0.775 (±0.012)	0.646 (±0.020)
Label - Sentence	0.782 (±0.011)	0.656 (±0.021)
Compound Score - Sentence	0.780 (±0.008)	0.651 (±0.018)
Compound Score ×1000 - Sentence	0.762 (±0.005)	0.617 (±0.008)
Negative Score - Sentence	0.777 (±0.012)	0.642 (±0.027)
Negative Score ×1000 - Sentence	0.774 (±0.013)	0.634 (±0.028)
Pos, Neg, Neu, Comp - Article	0.782 (±0.006)	0.659 (±0.017)
Pos, Neg, Neu, Comp ×1000 - Article	0.776 (±0.011)	0.652 (±0.019)
No Punctuation		
Sentiment Features	**Accuracy**	**F1 Score**
No Sentiment	**0.813 (±0.006)**	**0.722 (±0.006)**
Global Label - Article	0.766 (±0.006)	0.612 (±0.014)
Derived Label - Article	0.762 (±0.004)	0.596 (±0.008)
Compound Score - Article	0.769 (±0.014)	0.616 (±0.023)
Compound Score ×1000 - Article	0.755 (±0.011)	0.582 (±0.016)
Negative Score - Article	0.760 (±0.009)	0.596 (±0.029)
Negative Score ×1000 - Article	0.756 (±0.010)	0.591 (±0.018)
Label - Sentence	0.775 (±0.003)	0.624 (±0.008)
Compound Score - Sentence	0.768 (±0.009)	0.611 (±0.013)
Compound Score ×1000 - Sentence	0.762 (±0.004)	0.593 (±0.014)
Negative Score - Sentence	0.767 (±0.016)	0.618 (±0.032)
Negative Score ×1000 - Sentence	0.777 (±0.005)	0.635 (±0.013)
Pos, Neg, Neu, Comp - Article	0.770 (±0.008)	0.609 (±0.013)
Pos, Neg, Neu, Comp ×1000 - Article	0.767 (±0.003)	0.603 (±0.011)

Table 4.10: Accuracy and F1 scores of different sentiment feature embedding techniques. Classification is performed using the SVM classifier. Lowercasing of features and removal of punctuation is applied separately on the corpus features, all which are represented as ELMo Embeddings. Each of the listed scores is the mean and standard deviation of three identical tests.

with sentiment included rather than without - resulting in a more chaotic and unreliable system.

In the case of lowercased as well as non-punctuated features, sentence-level sentiment features seem to perform slightly better than article-level ones, suggesting the possibility of more granular sentiment being less invasive and distracting to the classifier. Moreover, scaling sentiment scores seems to be an obstruction to the classification, with a reduction in performance in nearly all cases.

Interestingly, results for experiments including sentiment hint towards better performance using lowercased features to those non-punctuated. Being the proven best generalised preprocessing approach back in Section 4.1, we think that lowercased features may be more tolerant across different classifiers, and able to interfere less than other data preprocessing with external features such as sentiment.

Differing to Anthonio and Kloppenburg (2019), we do not see any particular improvement from compound scores to negative. Both sentiment metrics seem to score in the same range of values along all test cases. Moreover, taking into consideration all VADER scores as suggested by Palić et al. (2019) does seem to improve on the classification performance compared to alternative article-level sentiment experiments, yet not by much. Indeed despite these reported improvements and the two systems using the same classification system that is the SVM, there is still no question that classification without sentiment, in our case, would be preferred.

4.3.1.2 | Sentiment as a Feature for Hyperpartisan News Classification - Discussion

Having thoroughly tested sentiment features embedded within hyperpartisan article textual features, the corresponding results are somewhat disappointing. As displayed in Table 4.10, among all experiments in which sentiment is involved both the accuracy and the F1 score suffered.

In calculating the F1 measure for each experiment, we noticed the same pattern throughout all embedding configurations; a high precision with very poor recall. Such consistently poor recall implies a high number of false negatives during classification. In our case this means a high number of article samples which in reality are hyperpartisan, being classified as neutral. Due to the low recall score, the F1 measure itself is decreased, while the accuracy score is reduced due to the number of articles wrongly classified as neutral.

Such an observation would suggest the hypothesis that the article sentiment could be masking further the already obscure nature of hyperpartisan articles. Indeed, it does seem from the results acquired that hyperpartisan news articles are not so easily distinguishable from their sentimental aspects, as one may initially assume.

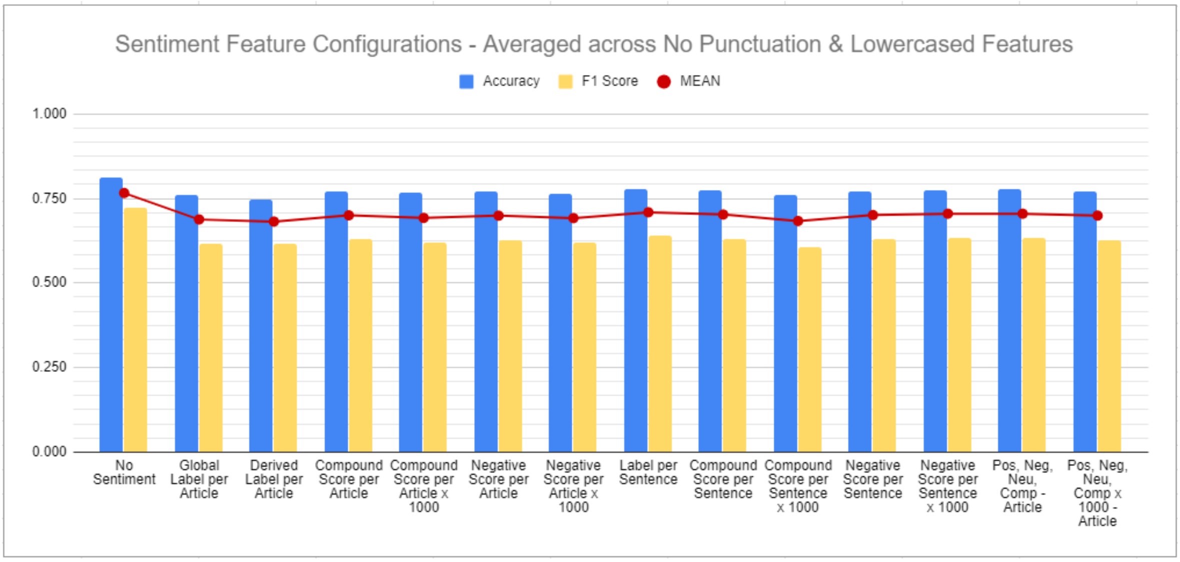


Figure 4.2: Average accuracy results for different sentiment embedding configurations averaged over no punctuation and lowercased textual features. Textual features are represented as ELMo embeddings and results are obtained for the SVM classifier.

In Figure 4.2 we plot the mean accuracy and F1 scores across both lowercased and non-punctuated features. A clear and distinguishable discrepancy in performance can be noticed between classification without any sentiment features, and any of the other experiments involving sentiment. Hence we conclude that from our testing the inclusion of sentiment with article features is not beneficial for the classification of hyperpartisan news articles, and thereby should not be included.

One must however ask why similar systems [Anthonio and Kloppenburg (2019); Palić et al. (2019)] report positive outcomes when incorporating sentiment. The answers as subjective to each system, with Anthonio and Kloppenburg (2019) focusing solely on the sentiment within the articles. The researchers only see an improvement in accuracy when VADER negative score is taken into account (as opposed to VADER compound). However with an accuracy of 0.5616, we and other systems achieving the same ranges of higher baseline performance than Anthonio and Kloppenburg (2019) do not experience the same levels of improvement.

In case of Palić et al. (2019), we notice that despite the improvements reported through the addition of sentiment, the authors achieve highest accuracy scores of 0.76128. With this accuracy the system is comparable to the performance of HyperPT when sentiment is included. Moreover, similar systems like Joo and Hwang (2019) mirror the reduction in performance encountered by our system during the above testing.

Different to the three systems above, we also represent our features as ELMo embeddings [Peters et al. (2018)]. This is a different approach to the feature representation techniques utilised by these studies and moreover, we are unsure whether in similar systems sentiment features are passed directly with textual features in the same way our system does or in a different manner, since this is not made particularly clear by the researchers.

These concerns could undoubtedly be explored further, and we do not believe to have explored all of the potential within the sentiment aspects of hyperpartisan news articles. Having said so, from the results above we conclude on leaving sentiment out of the final HyperPT system configuration, focusing on the textual article features themselves and individual feature elements which could be pivotal to the hyperpartisan nature of the article.

4.3.2 | Title and Body of a Hyperpartisan News Article

In trying to determine the versatility of the HyperPT classification system as well as gain further insight into the possible manifestations of hyperpartisan news articles, we perform experiments with the article title and article bodies of differing lengths. We perform training and testing on just the article titles and on the article body of different lengths - with and without the title. The corresponding test results can be examined in Table 4.11. The same seven experiments are performed on both lowercased and non-punctuated features (Section 4.1 and Section 4.2) - represented as ELMo embeddings and classified using an RBF kernel SVM.

4.3.2.1 | Evaluating Title and Body of a Hyperpartisan Article

Observing test results in Table 4.11, one notices that the article title does produce some interesting insights. Testing on solely the article title does, as may be expected, decrease the performance accuracy, yet surprisingly not by so much. Indeed it still does fairly well, achieving accuracies upwards of 70%. One however also notes that the corresponding F1 measure is significantly decreased to just upwards of 0.5 - implying decreased reliability and balance in the classifications.

Despite not being enough on its own to safely discard the article body, the title appears to enhance classification when appended to the body. The previously best scores of 0.813 (±0.006) [mean accuracy] and 0.722 (±0.006) [mean F1] achieved using non-punctuated features and full article body are exceeded by the same configuration with

Lowercase				
Features	**Mean Accuracy**	**Mean F1 Score**	**Highest Accuracy**	**Highest F1 Score**
Title	0.708 (±0.009)	0.561 (±0.013)	0.719	0.576
Title + Body (5 Sen)	0.756 (±0.004)	0.653 (±0.002)	0.760	0.654
Title + Body (15 Sen)	0.785 (±0.005)	0.697 (±0.010)	0.790	0.709
Title + Body (Full)	0.805 (±0.003)	0.732 (±0.005)	0.808	0.735
Body (5 Sen)	0.743 (±0.011)	0.628 (±0.018)	0.756	0.648
Body (15 Sen)	0.771 (±0.015)	0.679 (±0.030)	0.787	0.711
Body (Full)	0.808 (±0.013)	0.722 (±0.009)	0.809	0.730
No Punctuation				
Features	**Mean Accuracy**	**Mean F1 Score**	**Highest Accuracy**	**Highest F1 Score**
Title	0.718 (±0.008)	0.572 (±0.012)	0.725	0.585
Title + Body (5 Sen)	0.779 (±0.013)	0.684 (±0.021)	0.793	0.707
Title + Body (15 Sen)	0.800 (±0.012)	0.714 (±0.017)	0.813	0.732
Title + Body (Full)	**0.822 (±0.011)**	**0.745 (±0.012)**	**0.835**	**0.758**
Body (5 Sen)	0.754 (±0.004)	0.643 (±0.007)	0.758	0.649
Body (15 Sen)	0.774 (±0.005)	0.677 (±0.008)	0.780	0.681
Body (Full)	0.813 (±0.006)	0.722 (±0.006)	0.821	0.726

Table 4.11: Accuracy and F1 Scores using article title and body of multiple lengths. Features are represented as ELMo embeddings while classification is performed using SVM.

the addition of the article title. A mean accuracy of 0.822 (±0.011) and a mean F1 score of 0.745 (±0.012) is achieved, with the highest overall accuracy being that of 0.835 and the highest F1 of 0.758.

Concurrently with the article title, we test the article body at different lengths. Since it is processed as a series of sentences, we trim the article length by specifying the number of sentences to tolerate while removing the rest. Three different article lengths are examined; five sentences (simulating an introductory paragraph), fifteen sentences (roughly analogous to two large or three small paragraphs), and finally the entirety of the news article.

A correlation between the article length and the classification performance is noticeable, where accuracy and F1 measure scores increase gradually with the length of the article for any configuration tested. This may be since a lengthier article provides more data to be analysed. The addition of the title once again seems to improve the performance of any body length experiment - particularly visible for non-punctuated features, where we see an average of $\sim$ 1.8% increase in accuracy and a $\sim$ 3% increase in F1 score.

Just the same as observed with the article title, the classification performance on reduced article lengths, albeit degrading, is still maintained at a respectable range, even with such a small portion of the article being analysed as just five sentences; achieving mean accuracy scores of 0.756 (±0.004) and 0.779 (±0.013) respectively for lowercased and non-punctuated features.

4.3.2.2 | Title and Body of a Hyperpartisan Article - Discussion

One finds several systems incorporating the article title within the classification. Jiang et al. (2019) embed the article title features preceding to the body. Joo and Hwang (2019) take this work further by computing the sentiment of the article title and its cosine similarity to the article body. Systems such as Chen et al. (2019) alternatively take into consideration the length of the title, of the contained words and the number of capitalised words embedded within.

In considering the inclusion of the title within HyperPT, we proceeded with Jiang et al. (2019)'s simpler approach rather than more elaborated metrics. This is due to two reasons; the natural time limitations encompassing the study (discussed further in Chapter 5) and since research on alternative systems shows that the acquired performance does not justify the effort needed in implementing more elaborate techniques, rather than simply embedding the title with the body.

We display the results to our experimentations in Table 4.11. We feel that the unquestionable benefit observed from the addition of the article title is reasonable. Given that it is the introductory, crucial piece of text which readers skim to further peruse or dismiss an article, we are of the opinion that the main theme and narrative of the article would already be present in the title itself. Moreover, hyperpartisan news articles may attempt to capture the attention of individuals with a tendency to skim over news headlines through shocking or overly-dramatised titles. This is backed up by the reasonable accuracy results maintained when classifying solely the article title, divulging the distinguishing hyperpartisan elements already inside of the article title.

The same could be said on the length of the article. Despite the classification process being performed on a small portion of the article, reasonable accuracy results are maintained. Performance is further increased with the length of the article, yet taking into consideration the introductory few sentences of the article, one observes the already revealing hyperpartisan nature at the very beginning of the text.

From these insights we take away two important conclusions. The first one regards the pure classification aspects of the HyperPT system, where we include the article title and maintain the full length of the body due to the confirmed best performance in doing so. The second is on the nature of the hyperpartisan article itself, where we observe how very early on through the perusing of the text one is already seriously exposed to the inflammatory, biased and distinguishable hyperpartisan style of writing [Potthast et al. (2018)].

4.3.3 | Model Explainability and Saliency of Hyperpartisan Features

We analyse the classifier's decision-making to determine not only its behaviour but also the influence of each input feature within the classification. We do so by using LRP, a model explainability technique adapted from Vision to the area of textual NLP [Arras et al. (2016); Samek et al. (2017)], to retrace the steps from the output back through the classifier, till we reach the original input. As discussed with further detail in Section 2, LRP identifies which features are pivotal in supporting or opposing a prediction class through decomposition; redistributing the prediction function through the classifier's layers to eventually assign what we call a relevance score to each input variable.

Throughout this section, we repeat the same steps not only for the SVM classifier, but also for the CNN. This is since despite establishing better performance from the SVM classifier, we feel it would be a good opportunity to compare the two inherently different models in order to examine in which ways they are similar, and in which different.

In Section 4.3.3.1 we first evaluate the LRP algorithm against baseline techniques. We do so to determine the effectiveness and reliability of the algorithm before putting it to use on the hyperpartisan article samples. We then apply it on our classifiers and monitor the acquired results. The interpretability process is executed after the model is fully trained, and is performed by reinputting a number of input samples (article vectors) through the classifier, and capturing the outcome. LRP monitors the classification and provides each article input with a relevance score.

Finally in Section 4.3.3.3 we discuss the results acquired and derive our conclusions on the interpretability approach itself, the article features highlighted by this technique, what these results show us about the nature of the hyperpartisan article, and finally how can such features be leveraged for improved hyperpartisan news detection.

4.3.3.1 | Evaluating Layer-Wise Relevance Propagation

To evaluate LRP, we implement an approach inspired by Samek et al. (2017), where following the identification of the most salient features inside of an article, each feature is sequentially removed, from the most influential to the least. With each feature removal, classification is performed on the article, and the corresponding accuracy is monitored. This process is executed simultaneously on a number of articles, with the mean accuracy across the articles at each classification run being plotted.

The same process is repeated on alternative techniques; random feature removal and in the case of CNN, Sensitivity Analysis (SA). We expect the mean accuracy to drop the fastest and the furthest for the best detected salient features, since the removal of these features would imply the deterioration of the classification quality. Hence we can say that the interpretability process resulting in the heaviest accuracy deterioration is the best.

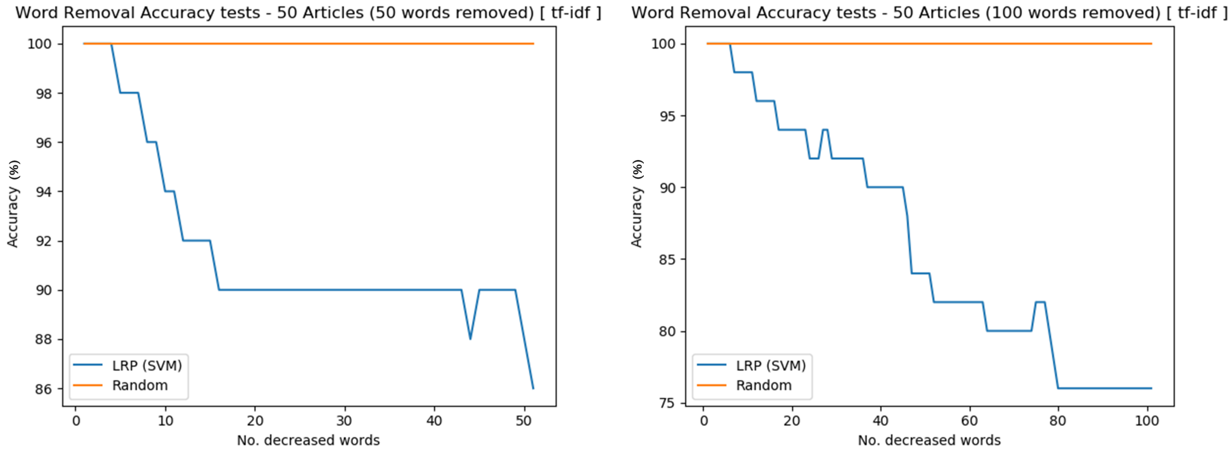


Figure 4.3: Evaluating the LRP interpretability algorithm by removing the 50 and 100 most salient features and monitoring the corresponding accuracy on the SVM classifier.

In a more ideal scenario we would have utilised SA as well on the SVM, but due to no direct way found for facilitating its implementation, the only possible way was to build the algorithm from first principles which, limited by time constraints, we decided not to opt for. Besides, having already developed the LRP algorithm for the SVM in similar fashion (Chapter 2), we believe that from the evaluation results below; against random feature removal in case of the SVM, and with the addition of SA for the CNN, the superior performance of LRP is clearly defined.

Moreover due to the Min-Max pooling of ELMo features before SVM classification (Chapter 2), we realised that it would be impossible to map back the aggregated and averaged vector representing the whole article to each individual feature. Therefore we decided on utilising TF-IDF representations which, after LRP rankings are issued, can be easily mapped to the corresponding textual features.

In Figure 4.3 one can observe the evaluation process of the LRP interpretability technique on the SVM classifier. Having ranked the features of 50 separate and correctly-classified articles, the top 50, and top 100 influential features for each article are successively removed. The accuracy starts from 100% only due to the purposely chosen articles being initially classified correctly. We hence see the gradual degradation of the SVM classification accuracy on these 50 articles with each removal of an influential feature.

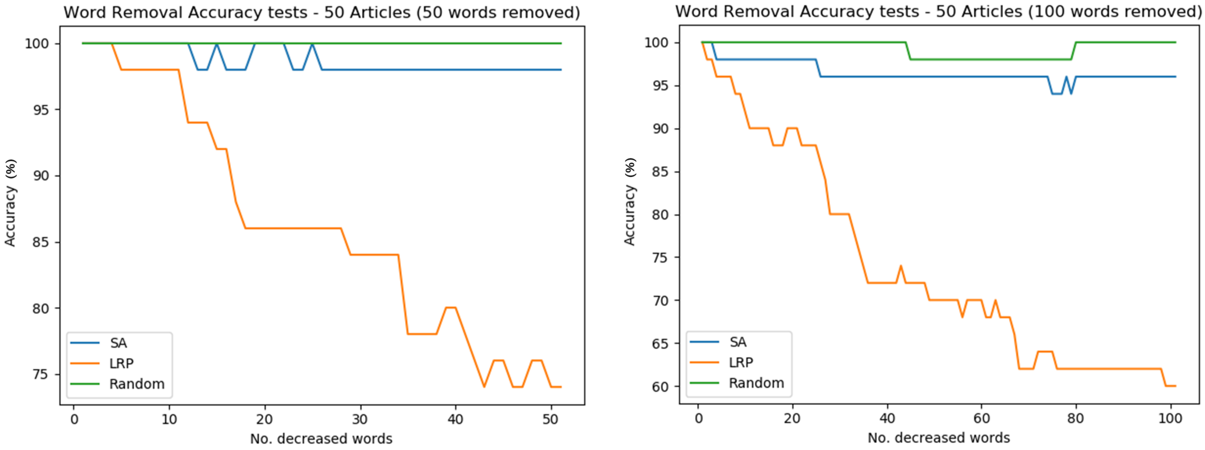


Figure 4.4: Evaluating the LRP interpretability algorithm by removing the 100 most salient features and monitoring the corresponding accuracy on the CNN classifier.

The same process can be observed for the CNN classifier (Figure 4.4), where in this case the accuracy degradation using LRP is also plotted against that using SA. As can be clearly observed from both Figure 4.3 and Figure 4.4, a large discrepancy in accuracy exists between the removal of LRP-flagged features and any of the other approaches. The poorest of the lot, as expected, would be random feature removal, where we randomly select and eliminate one of the article features. This is in no way an educated decision and therefore the likeliness of removing a highly salient feature is small.

Moreover, SA (in the case of CNN) performs just slightly better than random feature removal. This is not however unexpected, since similar outcomes were reported by published literature [Arras et al. (2016); Samek et al. (2017)]. Indeed the SA algorithm can be considered as a more primitive method compared to its counterpart - being based on the classifier's locally-evaluated gradient, while LRP redistributes the classifier's prediction back all the way through to the input. Moreover LRP is capable of detecting both features supporting a classification and features opposing it, while SA is only capable of the former.

With these observations we can confirm that using LRP on our classifiers does indeed provide us with the most salient input features making up the hyperpartisan news

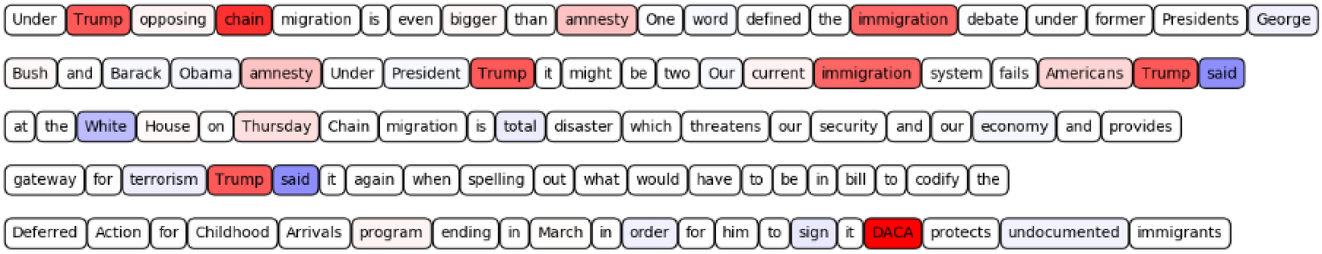

Under Trump opposing chain migration is even bigger than amnesty One word defined the immigration debate under former Presidents George
Bush and Barack Obama amnesty Under President Trump it might be two Our current immigration system fails Americans Trump said
at the White House on Thursday Chain migration is total disaster which threatens our security and our economy and provides
gateway for terrorism Trump said it again when spelling out what would have to be in bill to codify the
Deferred Action for Childhood Arrivals program ending in March in order for him to sign it DACA protects undocumented immigrants

Figure 4.5: A snippet of an article[2]correctly classified as hyperpartisan using the SVM classifier. The most salient features in the snippet are tagged using LRP. Red-tagged features support the Hyperpartisan classification while blue-tagged features oppose it.

articles. In Section 4.3.3.2 we now map back the rankings provided to the respective textual features such that we would be able to reconstruct and visually observe the influence of each feature.

4.3.3.2 | Determining the Saliency of Hyperpartisan Features

Having evaluated the LRP interpretability algorithm, we now analyse the saliency scores for the chosen articles. Following the LRP's saliency scores, we then map the feature vectors and corresponding scores back to the original words, reassembling the whole article.

Inspired by Arras et al. (2017)'s approach, we generate a heatmap of the article features by assigning a colour-code to each feature according to its influence value, with red implying that the feature supports hyperpartisanship while blue implies opposing it and supporting neutrality. White features are considered as not holding any specific influence within the classification. Moreover, the intensity of each feature colour determines the level of saliency the feature holds within the classification.

In Figure 4.5 and Figure 4.6 one observes snippets of two articles, one hyperpartisan and the other neutral, classified using the SVM classifier. As discussed in Section 4.3.3.1, features in this case are represented as a Bag-of-Words (BoW) of TF-IDF values. One must also note that due to the nature of the BoW representation and the inner-workings of the SVM, the same feature, no matter in which part of the article it is situated, is given the same influence - resulting in the same colour-code and intensity throughout all of the article. As we shall see further down, this differs when using the CNN classifier.

[2] ***Under Trump, opposing 'chain migration' is even bigger than 'amnesty'*** - `www.washingtonexaminer.com` [Last Accessed: 07-2020]
[3] ***President-Elect Trump Tours Washington*** - `www.nbcnews.com` [Last Accessed: 07-2020]

In the first article (Figure 4.5), one notices a lack of neutral features and two particular words flagged as being of hyperpartisan nature - repeatedly present throughout the article snippet. These two words are *Trump* and *immigration*, with two other hyperpartisan words being *chain* and *DACA*. With *Trump* and *DACA* both being proper nouns, we notice, throughout the rest of the article along with other, similar hyperpartisan cases, that the highest flagged hyperpartisan features tend to be proper nouns. Conversely, 'heavy' and superlative words such as *fails, disaster, threatens* and *terrorism* are not given such distinguishable influence scores.

The highlighting of proper nouns is also present in articles classified as neutral. In Figure 4.6 we find one such article. Indeed one can notice the persistence of the word *Trump* being flagged as hyperpartisan. Interestingly, despite being flagged as so, related words such as *President* and *Donald* are highlighted as neutral. From our analysis on similar articles bearing the same patterns, we propose a hypothesis as to why this, and similar cases, are so.

Articles which are typically of a neutral nature tend to report on high-profile individuals like the current president of the United States (being one of the most mentioned entities throughout our dataset) in full as a sign of authoritative respect - either as *President Donald Trump* or *President Trump*. Hyperpartisan articles, on the other hand, often attempt to criticise and ridicule the same individual, with the majority of the time referring to him simply as *Trump*. Hence the words *President* and *Donald* are more commonly found in articles of neutral nature, while *Trump* is widely used in hyperpartisan ones.

The rigid nature of hyperpartisan features classified using SVM is not shared with those classified using CNN. Indeed in the case of the CNN we are faced with a level of ambiguity within the article features, since in this case the context is also taken into

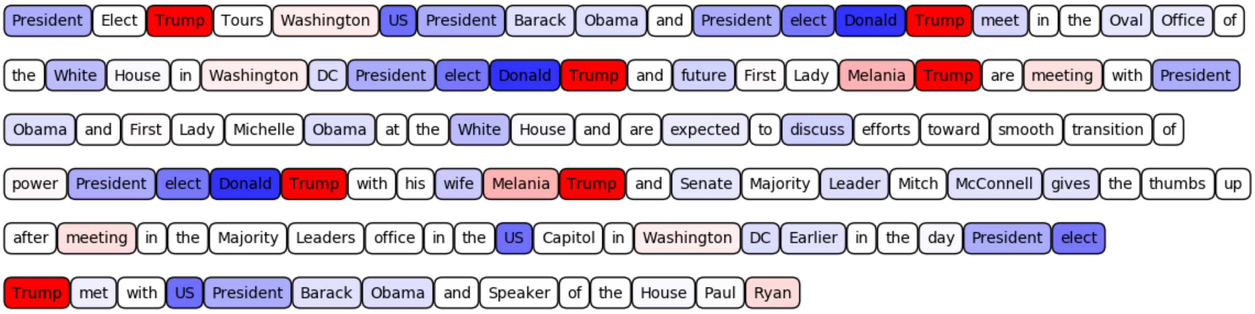
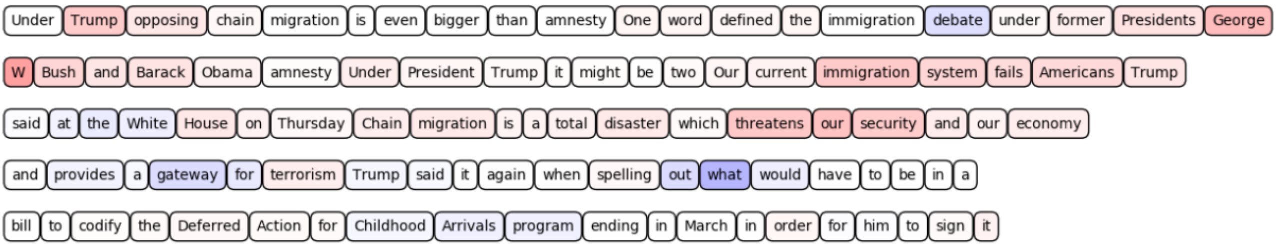
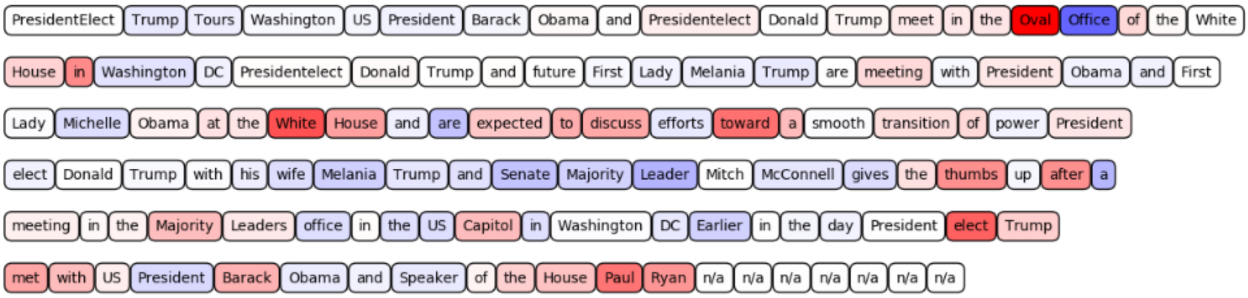

President Elect Trump Tours Washington US President Barack Obama and President elect Donald Trump meet in the Oval Office of
the White House in Washington DC President elect Donald Trump and future First Lady Melania Trump are meeting with President
Obama and First Lady Michelle Obama at the White House and are expected to discuss efforts toward smooth transition of
power President elect Donald Trump with his wife Melania Trump and Senate Majority Leader Mitch McConnell gives the thumbs up
after meeting in the Majority Leaders office in the US Capitol in Washington DC Earlier in the day President elect
Trump met with US President Barack Obama and Speaker of the House Paul Ryan

Figure 4.6: A snippet of an article[3]correctly classified as neutral using the SVM classifier. The most salient features in the snippet are tagged using LRP. Red-tagged features support the Hyperpartisan classification while blue-tagged features oppose it.



Under Trump opposing chain migration is even bigger than amnesty One word defined the immigration debate under former Presidents George
W Bush and Barack Obama amnesty Under President Trump it might be two Our current immigration system fails Americans Trump
said at the White House on Thursday Chain migration is a total disaster which threatens our security and our economy
and provides a gateway for terrorism Trump said it again when spelling out what would have to be in a
bill to codify the Deferred Action for Childhood Arrivals program ending in March in order for him to sign it

Figure 4.7: A snippet of an article[2] correctly classified as hyperpartisan using the CNN classifier. The most salient features in the snippet are tagged using LRP. Red-tagged features support the Hyperpartisan classification while blue-tagged features oppose it.



PresidentElect Trump Tours Washington US President Barack Obama and Presidentelect Donald Trump meet in the Oval Office of the White
House in Washington DC Presidentelect Donald Trump and future First Lady Melania Trump are meeting with President Obama and First
Lady Michelle Obama at the White House and are expected to discuss efforts toward a smooth transition of power President
elect Donald Trump with his wife Melania Trump and Senate Majority Leader Mitch McConnell gives the thumbs up after a
meeting in the Majority Leaders office in the US Capitol in Washington DC Earlier in the day President elect Trump
met with US President Barack Obama and Speaker of the House Paul Ryan n/a n/a n/a n/a n/a n/a n/a

Figure 4.8: A snippet of an article[3] correctly classified as neutral using the CNN classifier. The most salient features in the snippet are tagged using LRP. Red-tagged features support the Hyperpartisan classification while blue-tagged features opposite it.

consideration. Moreover, each feature can be given different influential scores for each occurrence within the article, since the influence score is not dependent solely on the word itself, but also on the neighbouring features.

Figure 4.7 and Figure 4.8 showcase the same two articles discussed above, classified and interpreted on the CNN classifier. As one can observe, the sharp, well defined features observed with the SVM are not present, yet we do notice that neighbouring clusters of features tend to share the same ranges of influence scores. This is particularly apparent in Figure 4.7, where the phrases *immigration system fails Americans* and *threatens our security* share the same ranges of hyperpartisan intensity. Furthermore, we notice that proper nouns such as *Trump*, which were previously flagged as extreme hyperpartisan, now vary according to where they are used.

Examining the neutral article featured in Figure 4.8, we notice more mixture of hypertpartisan and neutral features than in the case of the SVM (Figure 4.6). Not only so, but completely different sections of the article are highlighted, and where in the previous case we noticed solely the word *Trump* flagged as hyperpartisan, we now see various features which, according to the LRP interpretability scores, support the hyperpartisan side of the classification.

Hence indeed we notice an added layer of ambiguity and complexity with the CNN classifier not present with the simpler SVM. Moreover the two classifiers do not necessarily depend on the same features throughout the classification process, with the more complex and sensitive nature of the CNN possibly being a benefit or a hindrance to the system performance. This is discussed further in Section 4.3.3.3, where we reach our conclusions on the discussed results.

4.3.3.3 | Model Explainability and Saliency of Hyperpartisan Features - Discussion

Having examined in Section 4.3.3.2 the simpler, direct nature of the SVM on the article features, and the more complex, context-based approach by the CNN, one asks which approach would be the best?

In answering this question, we think that for the dataset presently used by this study, the SVM would be the best approach, both as shown by the classification accuracies and the clear, straight to the point feature interpretations observed in Section 4.3.3.2. However we understand the limitations of the small dataset used (Section 2.1) and the possibility of the CNN outperforming any traditional methods with larger datasets and more training data. This would definitely make for an interesting future project and addition to this work, as we remark further on in Chapter 5.

On another note, we compiled a list of the highest-scoring hyperpartisan and neutral features over 50 randomly-selected articles. Due to stopwords being present inside of the corpus and often times being flagged along with neighbouring features (in the case of context-sensitive influence for the CNN), we omitted such words from this list, focusing on more important, unique and interesting terms.

Observing the top ranked features in terms of saliency as featured in Table 4.12, one notices a number of proper nouns, particularly names of politically affiliated and well-known individuals, such as *Trump*, *Scott*, *Rauner*, *Hillary* and *Clinton*. Moreover we observe the multiple occurrences of important and often controversial entities such as *FBI* and *ISIS*.

Contrary to our initial hypotheses, inflammatory words such as superlative adverbs do not make the list for neither the SVM nor the CNN. This suggests an important possibility - that in classifying news articles for hyperpartisanship, both classification algorithms are more focused on the subject and the theme of the article rather than the way of writing. This then begs the question of whether topics similar to the ones upon

	SVM		**CNN**	
No.	**Hyperpartisan**	**Neutral**	**Hyperpartisan**	**Neutral**
1	*Trump*	*people*	*Trump*	*Trump*
2	*Conway*	*ISIS*	*York*	*said*
3	*Scott*	*Scott*	*said*	*Trumps*
4	*FBI*	*Clinton*	*Trumps*	*house*
5	*Rauner*	*Gaffney*	*President*	*new*
6	*emails*	*story*	*Hillary*	*Hillary*
7	*justice*	*Kimmel*	*Clinton*	*York*
8	*Fox*	*Brown*	*one*	*people*
9	*women*	*Fox*	*new*	*Republican*
10	*ISIS*	*Billy*	*times*	*according*
11	*Ive*	*victims*	*video*	*president*
12	*Berkeley*	*article*	*immigration*	*women*
13	*social*	*Korea*	*people*	*law*
14	*theyre*	*US*	*immigrants*	*wife*
15	*Hillary*	*said*	*state*	*2016*

Table 4.12: Top 15 hyperpartisan and neutral words calculated over a sample 50 randomly-selected, correctly-classified articles.

which the classifier is trained tend to be given the same label, and if new topics unseen by the classifier heavily risk the possibility of being classified wrongly.

Despite the possibility of this outcome, we are of the opinion that measures could be taken to minimise this risk. A larger and more diverse training dataset would first of all contain more diversity of topics. Moreover, we noticed that the dataset itself tends to contain a high number of American political themes. Due to the nature of the problem of hyperpartisanship, this is expected - yet with more diversity and room for training, the classifier is then forced to look for other features upon which to determine the hyperpartisanship of an article rather than letting itself settle on solely the entities within.

Furthermore, manually giving extra weight to the tone of writing of the article rather than simply the textual features within would incentivise the classification process to delegate more attention to these features, which coupled with a larger and more diverse dataset would compel the classifier to be more sensitive to otherwise overlooked features, before settling on the classification.

4.3.4 | External Features of Hyperpartisan News Articles - Discussion

In Section 4.3 we experimented with features not typically included in the initial stages of a classic classification process, but which are interesting to explore and hold potential for improving the baseline classification workflow.

We started off in Section 4.3.1 by experimenting with sentiment features. We tried a number of sentiment labelling and integration techniques within our articles, which despite the efforts, all resulted in a hindrance to the classification performance rather an improvement - compelling us to exclude sentiment from further analysis.

In Section 4.3.2 we then experimented with the length of the article body and the addition of the title. We found that the length of the title does affect the classification accuracy, with more length implying better performance. However we also noticed that even for extremely short article lengths, the overall classifier performance does not decrease by a large amount, implying that hyperpartisan elements are already present in the very initial few words of the article. This was further enforced by the addition of the title, where we noticed a performance increase in all experiment configurations.

Finally, we used the LRP model explainability algorithm on our hyperpartisan article instances with the aim of determining the most salient features affecting the classification process. This was addressed in Section 4.3.3, where we discussed how the most salient features inside of the articles tend to be individuals and entities rather than descriptive words, implying that the classifier is more preoccupied with the subject and entities within the article than the style of writing.

Having evaluated both the baseline classification process behind HyperPT, and external features which support further the classification, we close off Chapter 4 by finally comparing our system with the current state-of-the-art and winner of the 2019 SemEval Hyperpartisan News Articles challenge, Jiang et al. (2019).

4.4 | Evaluating HyperPT against the State-of-the-Art

In evaluating the HyperPT system, we established the SVM as the best performing data preprocessing and feature representation approaches for our data, with the CNN following not too far behind. Moreover we investigated the use of external features, namely; sentiment, article title, article length, and salient features - with the aim of exploiting these for the benefit of better detection and shedding more light on the nature of the hyperpartisan news article.

Throughout this section, we finally take all of this into consideration and compare the HyperPT system with the state-of-the-art. At the time of writing, the state-of-the-art and winner of the 2019 Hyperpartisan News Article challenge[4] is Jiang et al. (2019) - also known throughout the competition as Team Bertha Von Suttner.

4.4.1 | The State-of-the-Art in Hyperpartisan News Classification

Being a direct inspiration to this study, Jiang et al. (2019)'s proposed system shares a number of similarities with HyperPT. Features are represented as ELMo embeddings, yet different to our case, Jiang et al. (2019) average word vectors for each sentence, such that sentences are represented as singular vectors. In our case we maintain each corpus article as a set of consecutive word embeddings. We do so since albeit resulting in smaller volumes of data and faster training times, we think that averaging word vectors may to some extent counter the granularity provided by ELMo and thereby be more of a hindrance than an advantage to the classification performance.

The CNN classifier utilised by HyperPT is, as well, based on the architecture proposed by Jiang et al. (2019) - consisting of five parallel convolutional layers with kernel sizes of $2, 3, 4, 5$ and 6. These layers are followed by ReLU activation, batch normalisation and finally max-pooling before the output is forwarded to the fully-connected classification layers. Moreover Jiang et al. (2019) train multiple CNN models through 10-Fold Cross Validation, choosing the best three models to then be organised into an ensemble classification system, where the classifier results are averaged into one output label. More details on the CNN and its architecture can be found in Chapter 2.

4.4.2 | HyperPT vs the State-of-the-Art - Discussion

In comparing our system with that of Jiang et al. (2019), we couldn't evaluate the two systems directly due to two primary setbacks; a hidden test set used during the SemEval Hyperpartisan challenge is not made public, and hence we cannot use it to evaluate our system on the same test set. Moreover, sacrificing a chunk from the public training dataset and using it as a replacement for the original hidden one may result in an undertrained classifier due to the smaller size of the train set. This hence leaves us with one option, to compare the two systems based solely on the results achieved during the Cross Validation training phase.

[4] ***PAN SemEval Hyperpartisan News Detection (2019)*** - `www.pan.webis.de/semeval19` [Last Accessed: 05-2020]

Furthermore, the authors do not include whether the achieved training accuracy of 0.8404 is the mean over several tests or whether this is the highest accuracy achieved. We hence consider this both as the mean and the highest accuracy, despite knowing from our experience conducting similar tests, that it is unlikely for each experiment to achieve the same score. In doing so we however give more attention to the highest accuracy results achieved rather than the mean - since given this lack of clarity, we feel that it is a more fair comparison. In addition to this the authors also refrain from including the corresponding F1 measure, limiting us to solely the accuracy score.

Furthermore, we built Jiang et al. (2019)'s ensemble model as a separate implementation both to analyse and verify the classifier itself, comparing its performance with that of the HyperPT SVM and the state-of-the-art. Since as described above we do not hold any ownership over the SemEval hidden test set, we decided in the case of the ensemble CNN model to split the training set such that we have 20% of it being used as the test set. Similarly to Jiang et al. (2019), we train several CNN models using 10-Fold Cross Validation, and then select three of the best performing ones to evaluate their collective performance on the test set by averaging their classification outputs into one.

System Configuration	**Mean Accuracy**	**Mean F1**	**Highest Accuracy**	**Highest F1**
SVM (HyperPT)	0.822 (±0.011)	0.745 (±0.012)	0.835	0.758
Ens-CNN (HyperPT)	0.783 (±0.035)	0.740 (±0.057)	0.820	0.797
Ens-CNN (Jiang et al.)	**0.8404**	*N/A*	**0.8404**	*N/A*

Table 4.13: Mean and highest accuracy and F1 results for the HyperPT system and the state-of-the-art [Jiang et al. (2019)]. For the HyperPT system, we present the SVM classifier using RBF kernel and an Ensemble-CNN solution based on Jiang et al.'s system.

The corresponding results can be observed in Table 4.13. Focusing on the accuracies, the state-of-the-art exceeds any system proposed from our end. Our Ens-CNN model achieves a highest accuracy of 0.820, with an expectedly lower mean accuracy of 0.783 (±0.035). This could be attributed to the 20% less training data provided to the classifier compared to the state-of-the-art, however if one were to consider solely the highest accuracy score, the two systems score not too far apart.

Considering that it is an inherently different and simpler approach, the SVM actually manages to come close to the accuracy levels of the state-of-the-art. With a mean accuracy of 0.822 (±0.011), we are unsure of the actual equivalent mean accuracy by the state-of-the-art, however we feel that even if the mean is indeed as high as the 0.84 levels, the SVM would still have achieved close accuracy levels with a fraction of the training time and resources.

Between the highest result scored by the HyperPT SVM and the state-of-the-art, we find that there is an accuracy difference of 0.0054 (0.54%). This further enforces the point that the SVM fares considerably well. Indeed in the case of larger, more diverse datasets, as discussed earlier in Section 4.2, we do believe that the distance between the two classifier performances increases, with the SVM lacking behind since CNN-based solutions would be able to scale better to the larger volumes of data. However given the exact problem we are trying to tackle here - on the same dataset, we feel that our unique combination of ELMo embeddings adapted to the SVM classifier is a worthy alternative to the state-of-the-art.

Having presented these results, we feel it is not our place to decide on the best performing system between the two, but rather to propose an alternative and equally promising approach. At the end of the day before settling for a practical system to be used in a real-life scenario, other aspects come in play; the performance capabilities of the hardware upon which the system will perform, the size and diversity of the expected data, and the scaling potential of the system. An SVM-based system is significantly less resource-hungry than its DL counterpart, yet this comes at a cost of less performance and less scaling capabilities down the line. One must then decide based on his unique application which system would fit best.

4.5 | System Evaluation and Discussion - Summary

Throughout this chapter we evaluated the HyperPT system and its components, delving in detail into our evaluation approaches, the results achieved and the conclusions derived from these experiments.

We started off by evaluating the baseline components of our classification system. In Section 4.1 and Section 4.2 we respectively studied the best performing data preprocessing, feature representation and classification algorithms for the detection of hyperpartisan news articles. From the conducted experiments we concluded on feature lowercasing as the best generalised data preprocessing technique and ELMo embeddings as the best feature representation. Out of three classifiers, the SVM proved to be the best performer. We also determined that despite lowercasing of features being the best generalised data preprocessing technique, non-punctuated features tend to work notably better when coupled with the SVM.

Furthermore, we investigated the possibility of non-conventional features for further amplification of the classification performance - in doing so discovering further the hyperpartisan article itself. In Section 4.3 we investigated the addition of sentiment, the addition of the article title and experimented with the length of the article itself.

Moreover we then evaluated the LRP algorithm and used it to determine the saliency of features playing a pivotal role in the classification process.

Due to the disappointing performance of sentiment features we decided against maintaining them inside of the HyperPT system. Moreover we noticed a notable increase in performance when including the article title, maintaining a respectable accuracy score. Having been selected using LRP, the fifteen most salient features detailed from a sample of 50 randomly chosen articles include a number of proper nouns and names of powerful and politically affiliated individuals. We concluded from these observations that in classifying hyperpartisan news articles, more importance is given to the article subject and entities involved rather than the tone in which it is written.

Finally, we compared the entirety of the HyperPT system with the state-of-the-art; Jiang et al. (2019). We concluded that despite the state-of-the-art reporting higher accuracy results, the SVM-based system we proposed could be a worthy alternative, and depending on the application, it might also be preferred.

5

Conclusions

With the evaluation of the HyperPT system in Chapter 4 and the resulting discussions, we now give our conclusions on this study, where we presented our approach to the detection and classification of hyperpartisan news articles.

An introduction to the problem caused by hyperpartisan news was first given in Chapter 1, defining also HyperPT; our approach for the detection of hyperpartisan news articles. In the ensuing Chapter 2, we detailed the background behind techniques inspiring or directly used within our study - with a concise review of related work and published systems tackling the same or similar problems.

Having examined related literature, we moved on to Chapter 3, where our approach in designing the HyperPT system was examined. Here we discussed our design process, along with the physical implementation of the components making up the system. We summarised HyperPT into three main components; 1) data loading and preprocessing, 2) model classification and evaluation, and 3) model interpretability. Three classification algorithms were chosen as candidates for the detection of hyperpartisan news articles; the SVM, RF and CNN, before a detailed overview of the approach and implementation of the LRP interpretability algorithm was given.

In Chapter 4 we evaluated our methodology and its implementation. An elaborate system evaluation was performed, including a range of data preprocessing and feature representation techniques. Hyperparameter tuning was performed on each of the classifiers before they were thoroughly tested and compared with one another. The addition of sentiment features was examined and tested, resulting in an unexpected hindrance to the system performance. Furthermore, the consideration of the article title was noted as an important and effective addition to the system classification.

Finally, we evaluated the LRP model explainability algorithm and utilised it for the interpretation of the two most prominent classification models; the SVM and the CNN

- in doing so observing the resulting feature saliency scores. The best performer of the two, the SVM, was compared with the state-of-the-art, where we showed how our project fares as an alternative hyperpartisan classification system - featuring different advantages and characteristics to its competitor.

5.1 | Achieved Aims and Objectives

In conducting the HyperPT study, we aimed at addressing the five main objectives initially set in Chapter 1. Below we briefly discuss our observations and conclusions in tackling each objective.

5.1.1 | Features of a Hyperpartisan News Article

Through the LRP interpretability algorithm applied on the SVM and the CNN classifiers, each of the models' decision-making was observed. Consequently, a saliency score was given to each of the article features. In doing so we observed (Section 4.3.3) that due to the frequent occurrences of proper nouns and names of well-known individuals, the classification process tends to focus more on the subject of the article and the involved entities rather than its method of writing. This observation highlights the importance of entities within hyperpartisan news articles, however it also shows that the style of writing inside of the articles may not be the main priority, contrary to what was previously thought. We however tend to believe that with a larger and more diverse dataset, more generalisation would be present within the article features.

5.1.2 | Sentiment of a Hyperpartisan News Article

Despite mixed opinions found in related work on the addition of sentiment, tests conducted on our system and corresponding results implied a decrease in classification performance. Moreover, similar systems matching the same range of accuracies as us aligned with this conclusion. To extend the observations made in Section 5.1.1, we think that the lower importance given during classification to descriptive features compared to entities and subjects minimises the potential effects offered by sentiment derived from such descriptive features.

5.1.3 | Minimum length of text for an Article to be Hyperpartisan

As shown in Section 4.3.2, experiments were conducted with the article body, the article body extended by the title, and finally the title by itself. Moreover, we performed tests with different lengths of the article body; 5 sentences, 15 sentences, and full length. An instant increase in performance was noticed with the addition of the article title, suggesting the presence of hyperpartisan elements early on throughout the article. Furthermore, we noticed a direct correlation between the length of the article and the classification performance, implying that the longer the article, the better the probability of being classified correctly. This however does not rule out the possibility of short texts being hyperpartisan. Indeed even with just the title - the shortest length of text with which we experimented, the performance accuracy was maintained upwards of 0.70.

5.1.4 | Classifier for Hyperpartisan News Articles

Having evaluated the performance of the SVM, RF and CNN, the best performing of the three classifiers was the SVM. We hence compared the SVM with results reported by the state-of-the-art [Jiang et al. (2019)] and an implementation of the state-of-the-art ensemble CNN model built by ourselves. We found that testing the in-house implementation of the state-of-the-art achieves less performance than that reported by the researchers, though this can be attributed to the smaller dataset upon which it is trained, since the hidden SemEval Hyperpartisan News dataset was not made available to us. Moreover we noticed how the SVM achieves results close to those reported by the state-of-the-art with a difference in accuracy of 0.02 or 2%.

Considering the SVM's slightly poorer accuracy result yet notably faster execution time, we feel that the performance showcased by our system should put it in line as a worthy alternative hyperpartisan classification system to Jiang et al. (2019). We hence proposed several alternative advantages which one may consider in putting such a system to use; particularly the faster training times and less hardware requirements compared to the CNN-based state-of-the-art.

5.1.5 | Interpretation of the Classifier

Through the generation of feature saliency as discussed in Section 5.1.1, we noticed a tendency for the classifier associating entities with hyperpartisanship (and others with neutrality). We believe that despite the positive performance results, this tendency is not ideal, since with the introduction of new entities, topics and articles, the classifier may convey unprecedented behaviour. We think that this is mainly due to the small unvaried nature of the dataset (Section 2.1) upon which our classifiers are trained. We hence recommend a follow-up analysis of the classifiers' behaviour on a larger and more diverse dataset, as discussed further in Section 5.3.

5.2 | Critique and Limitations

Similar to any other system, HyperPT is not without its limitations. Despite these setbacks, we do feel that we attempted to present the best approach with the resources that were available, adding our contribution to the research community.

The first major limitation is the small size of the SemEval Hyperpartisan News Article dataset, particularly the By-Article collection, amounting to 645 articles (Section 2.1). Such a small size allowed us only a small range and variety of article samples, somewhat limiting the training of our classifiers. Moreover, we refrained from using the larger By-Publisher dataset, since as reported by a number of similar systems, the labelling of this collection is not of the same quality as that of the By-Article, potentially resulting in more of a hindrance to the research than an advantage (Section 2.1.2.1).

Coupled with this setback, we feel that the observations made from model interpretations (Section 4.3.3) suggest an important limitation to the proposed system. The tendency of assigning the highest influence scores to proper nouns and entities within the article texts implies a relationship between the entities and the article classification labels. We believe that the root of this issue may be directly related to the lack of variety within the small dataset of articles.

Finally, alternative feature representations and classification algorithms are always an interesting addition to the research conducted. With promising systems such as BERT word embeddings, the RNN classification architecture and the emerging transformer technology, we feel that the area of NLP and consequently the classification of hyperpartisan news still leaves ample room for further research. Due to the inevitable time constraints imposed on the project, we refrained from introducing further approaches, however we fully recommended such integrations as future work, as detailed in Section 5.3.

5.3 | Future Work

Further to the limitations discussed in Section 5.2, we feel that the proposed future work below would further extend the capabilities of the HyperPT system, consequently allowing for further research and more thorough observations into the topic of hyperpartisan news detection.

Being the main limitation encountered in conducting this study, we feel that a larger more diverse dataset would have allowed for more thorough and generalised training of the classifiers. As initially remarked in Section 4.3.3, the majority of the articles within the SemEval Hyperpartisan News Articles By-Article dataset collection revolve around the same theme of American politics. Despite this being expected since politics are often the subject of sensationalised and opinionated information, we feel that the extra effort taken in enlarging the existing dataset with the inclusion of different themes would greatly help in further training the classification algorithms.

As discussed in Section 2.1.2.1, one could look into de-noising and improving the otherwise poor labelling found within the By-Publisher data collection through article reclassification and using the insightful capabilities of Explainable A.I. (XAI). In doing so, the quality and reliability of the dataset could be improved, with the aim of then extending the smaller By-Article data collection with the new set. Moreover, to avoid the tendency of associating specific entities and subjects with hyperpartisanship, the manual introduction of weights to the stylistic features within the news articles would further encourage the detachment of the classifiers from entities, thereby implying more generalisation.

Furthermore, the addition and corresponding evaluation of promising feature representation and classification techniques would give us further insight into the problem of hyperpartisan news detection and perhaps suggest improved systems for tackling it. In particular, we feel that the consideration of the BERT word embedding technology would be an interesting extension to explore, with the implementation of RNN (LSTM) and transformer architectures as classifiers furthering the research conducted on Deep Learning classifiers for the problem at hand.

Finally, we think that the consideration of other external features such as the date of publication, the publishing entity and the article author may bring out more correlations to hyperpartisanship. Analysis of the article dissemination and the most prolific consumer audience may introduce patterns and observations on the vulnerability of different consumer communities and the corresponding spread of hyperpartisan content. The addition of such external features may aid both systems which like us classify hyperpartisan news according to the content within, and alternative approaches attempting to address this issue from different angles, such as the rate of spread and its prevention.

5.4 | Final Remarks

With a review of the system limitations in Section 5.2 and an overview of the recommended future work in Section 5.3, we give our final remarks on our classification system and the process which gradually but surely led us to developing the project in its entirety. The HyperPT project was a challenging yet fulfilling task, presenting various unique challenges during its research and development efforts.

All of the work involved is worth the effort when reflecting on the use such a system could have in today's hyper-communicated world, which, perhaps unknowingly, we find ourselves trying to adapt to on a regular basis. The damaging potential malicious content such as fake and hyperpartisan news could have on the masses is both large and easily uncontrollable - with unprecedented consequences. We hope that through the HyperPT system we pitch in our small contribution to the effort being made against such risks, with the aim of counteracting it and establishing control over its spread.

Finally, we hope that the work conducted throughout this project inspires the next generation of researchers and encourages more innovative systems to step up, furthering the state-of-the-art in both the area of hyperpartisan news detection, and the domain of textual NLP in general - one which harnesses great potential and which continues frequently see notable improvements.

References

Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

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

Rodrigo Agerri. Doris martin at semeval-2019 task 4: Hyperpartisan news detection with generic semi-supervised features. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 944–948, 2019.

Divyakant Agrawal, Ceren Budak, and Amr El Abbadi. Information diffusion in social networks: observing and affecting what society cares about. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 2609–2610. ACM, 2011.

Amal Alabdulkarim and Tariq Alhindi. Spider-jerusalem at semeval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 985–989, 2019.

Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, and Pieter-Jan Kindermans. innvestigate neural networks. *Journal of Machine Learning Research*, 20(93):1–8, 2019.

Shun-ichi Amari and Si Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6):783–789, 1999.

Evan Amason, Jake Palanker, Mary Clare Shen, and Julie Medero. Harvey mudd college at semeval-2019 task 4: The dx beaumont hyperpartisan news detector. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 967–970, 2019.

Talita Anthonio and Lennart Kloppenburg. Team kermit-the-frog at semeval-2019 task 4: Bias detection through sentiment analysis and simple linguistic features. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1016–1020, 2019.

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. Explaining predictions of non-linear classifiers in nlp. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 1–7, 2016.

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. " what is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12(8), 2017.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7), 2015.

Sameer Badaskar, Sachin Agarwal, and Shilpa Arora. Identifying real or fake articles: Towards better language modeling. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*, 2008.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *The Journal of Machine Learning Research*, 11: 1803–1831, 2010.

Vimala Balakrishnan and Ethel Lloyd-Yemoh. Stemming and lemmatization: a comparison of retrieval performances. 2014.

Hal Berghel. Malice domestic: The cambridge analytica dystopia. *Computer*, (5):84–89, 2018.

Yves Bestgen. Tintin at semeval-2019 task 4: Detecting hyperpartisan news article with only simple tokens. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1062–1066, 2019.

Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

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

Eric Brill. A simple rule-based part of speech tagger. Technical report, PENNSYLVANIA UNIV PHILADELPHIA DEPT OF COMPUTER AND INFORMATION SCIENCE, 1992.

Eric Brill. Some advances in transformation-based part of speech tagging. *arXiv preprint cmp-lg/9406010*, 1994.

Carole Cadwalladr and E Graham-Harrison. The cambridge analytica files. *The Guardian*, 21:6–7, 2018a.

Carole Cadwalladr and Emma Graham-Harrison. Revealed: 50 million facebook profiles harvested for cambridge analytica in major data breach. *The Guardian*, 17:2018, 2018b.

Eugene Charniak. Statistical techniques for natural language parsing. *AI magazine*, 18(4):33–33, 1997.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. 2013.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

Celena Chen, Celine Park, Jason Dwyer, and Julie Medero. Harvey mudd college at semeval-2019 task 4: The carl kolchak hyperpartisan news detector. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 957–961, 2019.

Gang Chen. A gentle tutorial of recurrent neural network with error backpropagation. 10 2016.

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

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

Rebekah Cramerus and Tatjana Scheffler. Team kit kittredge at semeval-2019 task 4: Lstm voting system. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1021–1025, 2019.

André Cruz, Gil Rocha, Rui Sousa-Silva, and Henrique Lopes Cardoso. Team fernando-pessa at semeval-2019 task 4: Back to basics in hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 999–1003, 2019.

Cedric De Boom, Steven Van Canneyt, Thomas Demeester, and Bart Dhoedt. Representation learning for very short texts using weighted word embedding aggregation. *Pattern Recognition Letters*, 80(C):150–156, 2016.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

Steven J DeRose. Grammatical category disambiguation by statistical optimization. *Computational linguistics*, 14(1), 1988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://www.aclweb.org/anthology/N19-1423`.

Thomas G Dietterich and Eun Bae Kong. Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. Technical report, Technical report, Department of Computer Science, Oregon State University, 1995.

Mehdi Drissi, Pedro Sandoval Segura, Vivaswat Ojha, and Julie Medero. Harvey mudd college at semeval-2019 task 4: The clint buchanan hyperpartisan news detector. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 962–966, 2019.

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

Michael Färber, Agon Qurdina, and Lule Ahmedi. Team peter brinkmann at semeval-2019 task 4: Detecting biased news articles using convolutional neural networks. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1032–1036, 2019.

Tristan Fletcher. Support vector machines explained. *Tutorial paper., Mar*, page 28, 2009.

Krishna Gade, Sahin Cem Geyik, Krishnaram Kenthapadi, Varun Mithal, and Ankur Taly. Explainable ai in industry. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3203–3204, 2019.

Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: a deep learning approach. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 513–520, 2011.

Randy Goebel, Ajay Chander, Katharina Holzinger, Freddy Lecue, Zeynep Akata, Simone Stumpf, Peter Kieseberg, and Andreas Holzinger. Explainable ai: the new 42? In *International cross-domain conference for machine learning and knowledge extraction*, pages 295–303. Springer, 2018.

Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *CoRR*, abs/1402.3722, 2014.

Priscilla E Greenwood and Michael S Nikulin. *A guide to chi-squared testing*, volume 280. John Wiley & Sons, 1996.

JP Gupta, Devendra K Tayal, and Arti Gupta. A tengram method based part-of-speech tagging of multi-category words in hindi language. *Expert Systems with Applications*, 38(12):15084–15093, 2011.

Viresh Gupta, Baani Leen Kaur Jolly, Ramneek Kaur, and Tanmoy Chakraborty. Clark kent at semeval-2019 task 4: Stylometric insights into hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 934–938, 2019.

Kazuaki Hanawa, Shota Sasaki, Hiroki Ouchi, Jun Suzuki, and Kentaro Inui. The sally smedley hyperpartisan news detector at semeval-2019 task 4. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1057–1061, 2019.

Susan C Herring. Computer-mediated discourse analysis. *Designing for virtual communities in the service of learning*, pages 338–376, 2004.

Lena Hettinger, Alexander Dallmann, Albin Zehe, Thomas Niebler, and Andreas Hotho. Claire at semeval-2018 task 7: Classification of relations using embeddings. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 836–841, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, 2018.

Clayton J Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth international AAAI conference on weblogs and social media*, 2014.

Tim Isbister and Fredrik Johansson. Dick-preston and morbo at semeval-2019 task 4: Transfer learning for hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 939–943, 2019.

Ye Jiang, Johann Petrak, Xingyi Song, Kalina Bontcheva, and Diana Maynard. Team bertha von suttner at semeval-2019 task 4: Hyperpartisan news detection using elmo sentence representation convolutional network. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 840–844, 2019.

Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972.

Karen Sparck Jones and Peter Willett. *Readings in information retrieval*. Morgan Kaufmann, 1997.

Youngjun Joo and Inchon Hwang. Steve martin at semeval-2019 task 4: Ensemble learning model for detecting hyperpartisan news. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 990–994, 2019.

P. Joshi, J. Hearty, B. Sjardin, L. Massaron, and A. Boschetti. *Python: Real World Machine Learning*, pages 621–623. Packt Publishing, 2016. ISBN 9781787120679. URL `https://books.google.com.mt/books?id=g57cDgAAQBAJ`.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. Semeval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, 2019.

Yoon Kim. Convolutional neural networks for sentence classification. In *EMNLP*, 2014.

Jürgen Knauth. Orwellian-times at semeval-2019 task 4: A stylistic and content-based classifier. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 976–980, 2019.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th International Conference on Data Mining*, pages 1103–1108. IEEE, 2013.

Sebastian Lapuschkin, Alexander Binder, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. The lrp toolbox for artificial neural networks. *The Journal of Machine Learning Research*, 17(1):3938–3942, 2016.

Alberto Lavelli, Fabrizio Sebastiani, and Roberto Zanoli. Distributional term representations: an experimental comparison. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 615–624, 2004.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Nayeon Lee, Zihan Liu, and Pascale Fung. Team yeon-zi at semeval-2019 task 4: Hyperpartisan news detection by de-noising weakly-labeled data. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1052–1056, 2019.

Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70, 2002.

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

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. 1993.

Donald Michie, David J Spiegelhalter, CC Taylor, et al. Machine learning. *Neural and Statistical Classification*, 13(1994):1–298, 1994.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013b.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

Rodrigo Moraes, JoãO Francisco Valiati, and Wilson P GaviãO Neto. Document-level sentiment classification: An empirical comparison between svm and ann. *Expert Systems with Applications*, 40(2):621–633, 2013.

José G Moreno, Yoann Pitarch, Karen Pinel-Sauvagnat, and Gilles Hubert. Rouletabille at semeval-2019 task 4: Neural network baseline for identification of hyperpartisan publishers. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 981–984, 2019.

Osman Mutlu, Ozan Arkan Can, and Erenay Dayanik. Team howard beale at semeval-2019 task 4: Hyperpartisan news detection with bert. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1007–1011, 2019.

Bang Nguyen, Xiaoyu Yu, TC Melewar, and Junsong Chen. Brand innovation and social media: Knowledge acquisition from social media, market orientation, and the moderating role of social media strategic capability. *Industrial Marketing Management*, 51:11–25, 2015.

Duc-Vu Nguyen, Thin Dang, and Ngan Nguyen. Nlp@ uit at semeval-2019 task 4: The paparazzo hyperpartisan news detector. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 971–975, 2019.

Zhiyuan Ning, Yuanzhen Lin, and Ruichao Zhong. Team peter-parker at semeval-2019 task 4: Bert-based method in hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1037–1040, 2019.

Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. 11 2015.

Edgar Osuna, Robert Freund, and Federico Girosit. Training support vector machines: an application to face detection. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pages 130–136. IEEE, 1997.

Niko Palić, Juraj Vladika, Dominik Čubelić, Ivan Lovrenčić, Maja Buljan, and Jan Šnajder. Takelab at semeval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 995–998, 2019.

Georgios Paltoglou and Mike Thelwall. Twitter, myspace, digg: Unsupervised sentiment analysis in social media. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(4):1–19, 2012.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86, 2002.

Olga Papadopoulou, Giorgos Kordopatis-Zilos, Markos Zampoglou, Symeon Papadopoulos, and Yiannis Kompatsiaris. Brenda starr at semeval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 924–928, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

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

Carla Pérez-Almendros, Luis Espinosa Anke, and Steven Schockaert. Cardiff university at semeval-2019 task 4: Linguistic features for hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 929–933, 2019.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL `https://www.aclweb.org/anthology/N18-1202`.

Slav Petrov, Dipanjan Das, and Ryan McDonald. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*, 2011.

Martin F Porter. Snowball: A language for stemming algorithms, 2001.

Martin F Porter et al. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. A stylometric inquiry into hyperpartisan and fake news. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 231–240, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1022. URL `https://www.aclweb.org/anthology/P18-1022`.

Laura Elena Raileanu and Kilian Stoffel. Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, 41(1):77–93, 2004.

Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing*, 1996.

Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.

Victoria Rubin, Niall Conroy, Yimin Chen, and Sarah Cornwell. Fake news or truth? using satirical cues to detect potentially misleading news. In *Proceedings of the Second Workshop on Computational Approaches to Deception Detection*, pages 7–17, 2016.

Victoria L Rubin, Niall J Conroy, and Yimin Chen. Towards news verification: Deception detection methods for news discourse. 2015.

Abdelrhman Saleh, Ramy Baly, Alberto Barrón-Cedeño, Giovanni Da San Martino, Mitra Mohtarami, Preslav Nakov, and James Glass. Team QCRI-MIT at SemEval-2019 task 4: Propaganda analysis meets hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1041–1046, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/S19-2182. URL `https://www.aclweb.org/anthology/S19-2182`.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. 2017.

Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

Saptarshi Sengupta and Ted Pedersen. Duluth at semeval-2019 task 4: The pioquinto manterola hyperpartisan news detector. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 949–953, 2019.

Daniel Shaprin, Giovanni Da San Martino, Alberto Barrón-Cedeño, and Preslav Nakov. Team jack ryder at semeval-2019 task 4: Using bert representations for detecting hyperpartisan news. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1012–1015, 2019.

Vertika Srivastava, Ankita Gupta, Divya Prakash, Sudeep Kumar Sahoo, RR Rohit, and Yeon Hyang Kim. Vernon-fenwick at semeval-2019 task 4: Hyperpartisan news detection using lexical and semantic features. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1078–1082, 2019.

Bozhidar Stevanoski and Sonja Gievska. Team ned leeds at semeval-2019 task 4: Exploring language indicators of hyperpartisan reporting. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1026–1031, 2019.

James Surowiecki. Wisdom of crowds: The wisdom of crowds, 2004.

Marcella Tambuscio, Giancarlo Ruffo, Alessandro Flammini, and Filippo Menczer. Fact-checking effect on viral hoaxes: A model of misinformation spread in social networks. In *Proceedings of the 24th international conference on World Wide Web*, pages 977–982. ACM, 2015.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*, 2019.

Peter D Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. 2002.

Udo Undeutsch. Beurteilung der glaubhaftigkeit von aussagen. *Handbuch der psychologie*, 11:26–181, 1967.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Andrew Wiese, Valerie Ho, and Emily Hill. A comparison of stemmers on source code identifiers for software search. In *2011 27th IEEE International Conference on Software Maintenance (ICSM)*, pages 496–499. IEEE, 2011.

W John Wilbur and Karl Sirotkin. The automatic identification of stop words. *Journal of information science*, 18(1):45–55, 1992.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, 26(3):1–37, 2008.

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

Jiacheng Xu, Danlu Chen, Xipeng Qiu, and Xuan-Jing Huang. Cached long short-term memory neural networks for document-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1660–1669, 2016.

Shailesh Kumar Yadav. Sentiment analysis and classification: a survey. *International Journal of Advance Research in Computer Science and Management Studies*, 3(3):113–121, 2015.

Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 25–26. Association for Computational Linguistics, 2007.

Chia-Lun Yeh, Babak Loni, and Anne Schuth. Tom jumbo-grumbo at semeval-2019 task 4: Hyperpartisan news detection with glove vectors and svm. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1067–1071, 2019.

Yichun Yin, Yangqiu Song, and Ming Zhang. Document-level multi-aspect sentiment classification as machine comprehension. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2044–2054, 2017.

Albin Zehe, Lena Hettinger, Stefan Ernst, Christian Hauptmann, and Andreas Hotho. Team xenophilius lovegood at semeval-2019 task 4: Hyperpartisanship classification using convolutional neural networks. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1047–1051, 2019.

Chiyu Zhang, Arun Rajendran, and Muhammad Abdul-Mageed. Ubc-nlp at semeval-2019 task 4: Hyperpartisan news detection with attention-based bi-lstms. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1072–1077, 2019.

Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253, 2018.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. Attention-based lstm network for cross-lingual sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 247–256, 2016.