Clickbait detection using multimodel fusion and transfer learning

Rajapaksha Waththe Vidanelage Praboda Chathurangani Rajapaksha

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2020IPPAS025

Thèse de doctorat

Clickbait Detection using Multimodel Fusion and Transfer Learning

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom SudParis

École doctorale n°626 Ecole Doctorale de l'Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Evry, le 27/11/2020, par

RAJAPAKSHA WATHTHE VIDANELAGE PRABODA CHATHURANGANI RAJAPAKSHA

Composition du Jury :

Gareth Tyson Maitre de Confèrences, Queen Mary University of London - UK	Président
Xiaoming Fu Professeur , University of Goettingen - Germany	Rapporteur
Christophe Cerisara Chercheur, CNRS - France	Rapporteur
Bruce Maggs Professeur, Duke University - USA	Examinateur
Cecile Bothorel Maitre de Confèrences, IMT Atlantique - France	Examinateur
Gareth Tyson Maitre de Confèrences, Queen Mary University of London - UK	Examinateur
Noel Crespi Professeur, IMT, Telecom SudParis - France	Directeur de thèse
Reza Farahbakhsh Maitres de Conférences associé, IMT, Telecom SudParis - France	Co-directeur de thèse

DOCTORAL SCHOOL

Title : Clickbait Detection using Multimodel Fusion and Transfer Learning

Keywords : Clickbait, Transfer Learning, BERT, XLNet, RoBERTa, Deep Learnig, Sentiment Classification, Topic Detection, Originality Detection, Social Media, Facebook, Twitter, News Media

Abstract : Internet users are likely to be victims to clickbaits assuming as legitimate news. The notoriety of clickbait can be partially attributed to misinformation as Internet users are likely to be victims to clickbaits assuming as legitimate news. The notoriety of clickbait can be partially attributed to misinformation as clickbaits use an attractive headline that is deceptive, misleading or sensationalized. A major type of clickbait is in the form of spam and advertisements which is used to redirect users to web sites that sells products or services (often of dubious quality). Another common type of clickbait is designed to appear as news headlines and redirect readers to their online venues intending to make revenue from page views, but these news can be deceptive, sensationalized and misleading. News media often use clickbaits to propagate news using a headline which lacks greater context to represent the article. Since news media exchange information by acting as both content providers and content consumers, misinformation that is deliberately created to mislead requires serious attention. Hence, an automated mechanism is required to explore likelihood of a news item being clickbait. Predicting how clickbaity a given news item is difficult as clickbaits are very short messages and written in obscured way. The main feature that can be used to identify clickbait is to explore the gap between what is promised in the social media post, news headline and what is delivered by the article linked from it. The recent enhancement to Natural Language Processing (NLP) can be adapted to distinguish linguistic patterns and syntaxes among social media post, news headline and news article. In my Thesis, I propose two innovative approaches to explore clickbaits generated by news media in social media. Contributions of my Thesis are two-fold : 1) propose a multimodel fusion-based approach by incorporating deep learning and text mining techniques, and 2) adapt Transfer Learning (TL) models to investigate the efficacy of transformers for predicting clickbait contents.

In the first contribution, the fusion model is built on using three main features, namely similarity between post and headline, sentiment of the post and headline, and topical similarity between news article and post. The fusion model uses three different algorithms to generate output for each feature mentioned above and fuse them at the output to generate the final classifier. In addition to implementing the fusion classifier, we conducted four extended experiments mainly focusing on news media in social media. The first experiment is on exploring content originality of a social media post by amalgamating the features extracted from author's writing style and her online circadian rhythm. This originality detection approach is used to identify news dissemination patterns among news media community in Facebook and Twitter by observing news originators and news consumers. For this experiment, dataset is collected using our implemented crawlers through Facebook Graph API and Twitter streaming APIs. The next experiment is on exploring flaming events on news media in Twitter by using an improved sentiment classification model. The final experiment is focused on detecting topics that are discussed in a real-time meeting with the aim of generating a brief summary at the end.

The second contribution is to adapt Transfer Learning models for the clickbait detection task. We evaluated performances of three Transfer Learning models (BERT, XLNet and RoBERTa), and delivered a set of architectural changes to optimize these models. We believed that these three models are the representatives of most of the other Transfer Learning models in terms of their architectural properties (Autoregressive model vs Autoencoding model) and training datasets. The experiments are conducted by introducing advanced fine-tuning approaches to each model such as layer pruning, attention pruning, weight pruning, model expansion and generalization. To the best of authors' knowledge, there have been an insignificant number of attempts to use Transfer Learning models on clickbait detection tasks and no any comparative analysis of multiple Transfer Learning models focused on this task.

Titre : Détection de Clickbait utilisant Fusion Multimodale et Apprentissage par Transfert

Mots clés : Clickbait, Apprentissage par transfert, BERT, XLNet, RoBERTa, l'apprentissage en profondeur, Classification des sentiments, Détection de sujets, Détection d'originalité, Médias sociaux, Facebook, Twitter, Médias d'information

Résumé : Presque tous les internautes sont susceptibles d'être victimes de clickbait, suppposant à tort qu'il s'agit d'informations légitimes. Un type important de clickbait se présente sous la forme de spam et de publicités qui sont utilisés pour rediriger les utilisateurs vers des sites web. Un autre type de "clickbait" est conçu pour faire la une des journaux et rediriger les lecteurs vers leurs sites en ligne, mais ces nouvelles sensasionelles peuvent être trompeuses. Il est difficile de prédire le degré de click-baity d'une nouvelle donnée car les clickbait sont des messages très courts et écrits de manière souvent obscure. La principale caractéristique qui permet d'identifier les clickbait est d'explorer l'écart entre ce qui est attendu dans un post, le titre de l'information et l'information réellement présente dans l'article qui y est lié. Dans cette thèse, on propose deux approches innovantes pour explorer le clickbait généré par les médias d'information dans les médias sociaux. Les contributions 1) de proposer une approche multimodèle basée sur la fusion en incorporant des techniques d'apprentissage profond et d'exploration de texte et 2) d'adapter les modèles d'apprentissage par transfert (TL) pour étudier l'efficacité des transformateurs permettant de prédire le contenu des clickbaits.

Dans la première contribution, le modèle de fusion repose sur l'utilisation de trois caractéristiques principales, à savoir la similitude entre la publication et le titre, le sentiment de la publication et du titre, et la similitude d'actualité entre l'article de presse et la publication. Le modèle de fusion utilise trois algorithmes différents pour générer une sortie pour chaque caractéristique mentionnée ci-dessus et les fusionner à la sortie pour générer le classificateur final. En plus de la mise en œuvre du classificateur de fusion, nous avons mené quatre expériences étendues principalement axées sur les médias d'information dans les médias sociaux. La première expérience consiste à explorer l'originalité du contenu d'une publication sur les réseaux sociaux en fusionnant les caractéristiques extraites du style d'écriture de l'auteur et de son rythme circadien en ligne. Cette approche de détection de l'originalité est utilisée pour identifier les modèles de diffusion des nouvelles parmi la communauté des médias d'information sur Facebook et Twitter en observant les auteurs et les consommateurs de nouvelles. Pour cette expérience, l'ensemble de données est collecté à l'aide de nos robots d'exploration implémentés via l'API Facebook Graph et les API de streaming Twitter. La prochaine expérience consiste à explorer les événements enflammés sur les médias d'information sur Twitter en utilisant un modèle de classification des sentiments amélioré. L'expérience finale se concentre sur la détection de sujets discutés lors d'une réunion en temps réel dans le but de générer un bref résumé à la fin.

La deuxième contribution est d'adapter les modèles de Transfer Learning pour la tâche de détection de clickbait. Nous avons évalué les performances de trois modèles d'apprentissage par transfert (BERT, XLNet et RoBERTa) et fourni un ensemble de modifications architecturales pour optimiser ces modèles. Nous pensons que ces trois modèles sont les représentants de la plupart des autres modèles d'apprentissage par transfert en termes de propriétés architecturales (modèle autorégressif vs modèle d'autoencodage) et de jeux de données de formation. Les expériences sont menées en introduisant des approches avancées de réglage fin pour chaque modèle telles que l'élagage des couches, l'élagage à l'attention, l'élagage au poids, l'expansion et la généralisation du modèle. À la connaissance des auteurs, il y a eu un nombre insignifiant de tentatives d'utilisation des modèles de transfert d'apprentissage sur les tâches de détection de clickbait et aucune analyse comparative de plusieurs modèles de transfert d'apprentissage axés sur la tâche de détection de clickbait.

Institut Polytechnique de Paris
91120 Palaiseau, France

Doctor of Philosophy (PhD) Thesis
Institut-Mines Télécom, Télécom SudParis
& Institut Polytechnique de Paris (IP Paris)

Specialization

Computer Science

presented by

Rajapaksha Waththe Vidanelage
Praboda Chathurangani Rajapaksha

Clickbait Detection using Multimodel Fusion and Transfer Learning

Committee:

Xiaoming Fu	Reviewer	Professor, University of Goettingen - Germany
Christophe Cerisara	Reviewer	Researcher, CNRS - France
Bruce Maggs	Examiner	Professor, Duke University - USA
Cecile Bothorel	Examiner	Senior Lecturer, IMT Atlantique - France
Gareth Tyson	Examiner	Senior Lecturer, Queen Mary University of London - UK
Noel Crespi	Advisor	Professor, IMT, Telecom SudParis - France
Reza Farahbakhsh	Co-supervisor	Adjunct Assistant Professor, IMT, Telecom SudParis - France

Thèse de Doctorat (PhD) de
Institut-Mines Télécom, Télécom SudParis
et l'Institut Polytechnique de Paris (IP Paris)

Spécialité

Informatique

présentée par

Rajapaksha Waththe Vidanelage
Praboda Chathurangani Rajapaksha

Détection de Clickbait utilisant Fusion Multimodale et Apprentissage par Transfert

Jury composé de :

Xiaoming Fu	Rapporteur	Professeur, University of Goettingen - Germany
Christophe Cerisara	Rapporteur	Chercheur, CNRS - France
Bruce Maggs	Examinateur	Professeur, Duke University - USA
Cecile Bothorel	Examinateur	Maitre de Confèrences, IMT Atlantique - France
Gareth Tyson	Examinateur	Maitre de Confèrences, Queen Mary University of London - UK
Noel Crespi	Directeur de thèse	Professeur, IMT, Telecom SudParis - France
Reza Farahbakhsh	Co-encadrant	Maitres de Conférences Associé, IMT, Telecom SudParis - France

Dedication

To My Family

Acknowledgements

First and foremost, I would like to express my gratitude and appreciation for Prof. Noel Crespi who gave me the opportunity to do this research work and provided me all the guidance and support. I really admire him for the given freedom to conduct independent research. I always enjoyed working with him and it increased my exposure and knowledge in many aspects.

I would like to thank my co-supervisor Dr. Reza Farahbakhsh who has supported me throughout my research. He continuously provided encouragement and was always willing and enthusiastic to assist in any way at anytime. Most importantly, he provided advices on every entangled situation and motivated whenever I lacked inspiration.

I wish to express my sincere gratitude to my thesis reviewers, Prof. Xiaoming Fu and Dr. Christophe Cerisara for their useful reviews and suggestions, which helped me to improve the quality of my thesis. A special thank to Prof. Bruce Maggs, Dr. Gareth Tyson and Prof. Cecile Bothorel for being the part of my jury as examiners for my thesis defense.

A very special thank goes Uva Wellassa University, Sri Lanka for granting permission to pursue my PhD and extending my leave period until the completion of my PhD.

My special thanks to all the lovely team members of The Data Intelligence and Communication Engineering Lab, Telecom SudParis, especially Yasir and Samin (my closest friends at the TSP), Ibrahim, Marzieh, Koosha, Faraz, Amir, Nikesh, Bahram, Shohreh, Hamza and Shanay. The time spent with you guys were un-forgettable and will always miss hangouts, tea parties and birthday parties with you all. Thanks to Prof. Roberto who always gave useful suggestions and comments while doing projects and appreciate your dedication on those. Special thank goes to Valerie Mateus, the secretary of RS2M Department. She was always very kind and generous in solving tedious administrative tasks. A deep thanks to Veronique Guy, the administrative responsible of PhD program, who always helped me a lot in dealing with PhD administrative tasks.

I cannot begin to express my thanks to my best friend with whom I shared my sadness, happiness and thanks for listening, offering me advices, caring for me, supporting me through this entire process and being there whenever I needed a friend.

My profound love, respect and thank goes to my family members : my father (Thaththi), my mother (Ammi), my sister, my brother, mother-in-law, father-in-law and sister-in-law. They always prayed for me, supported me and encouraged me to achieve this new and hard milestone of my life. My gratitude goes to Thaththi and Ammi for identifying my strengths and weaknesses during my childhood and permitting me to do whatever I love to do. Thank you for believing in me and supporting me, and without you I would not have made it through my PhD.

Last but not least, my biggest thanks to my husband Suranga Jayasingha, who always care for me and tolerating all the hardships and providing continuous support during my PhD. Thank you for the countless times you went out of your way to make sure I was comfortable and for all the compromises. You are truly a wonderful husband and thank you is a small word for all that you have done for me.

I hope they find here the expression of my deep gratitude and appreciation.

Praboda Rajapaksha
27^{th} November 2020

Abstract

Internet users are likely to be victims to clickbaits assuming as legitimate news. The notoriety of clickbait can be partially attributed to misinformation as clickbaits use an attractive headline that is deceptive, misleading or sensationalized. A major type of clickbait is in the form of spam and advertisements which is used to redirect users to web sites that sells products or services (often of dubious quality). Another common type of clickbait is designed to appear as news headlines and redirect readers to their online venues intending to make revenue from page views, but these news can be deceptive, sensationalized and misleading. News media often use clickbaits to propagate news using a headline which lacks greater context to represent the article. Since news media exchange information by acting as both content providers and content consumers, misinformation that is deliberately created to mislead requires serious attention. Hence, an automated mechanism is required to explore likelihood of a news item being clickbait. Predicting how clickbaity a given news item is difficult as clickbaits are very short messages and written in obscured way. The main feature that can be used to identify clickbait is to explore the gap between what is promised in the social media post, news headline and what is delivered by the article linked from it. The recent enhancement to Natural Language Processing (NLP) can be adapted to distinguish linguistic patterns and syntaxes among social media post, news headline and news article. In my Thesis, I propose two innovative approaches to explore clickbaits generated by news media in social media. Contributions of my Thesis are two-fold: 1) propose a multimodel fusion-based approach by incorporating deep learning and text mining techniques, and 2) adapt Transfer Learning (TL) models to investigate the efficacy of transformers for predicting clickbait contents.

In the first contribution, the fusion model is built on using three main features, namely similarity between post and headline, sentiment of the post and headline, and topical similarity between news article and post. The fusion model uses three different algorithms to generate output for each feature mentioned above and fuse them at the output to generate the final classifier. In addition to implementing the fusion classifier, we conducted four extended experiments mainly focusing on news media in social media. The first experiment is on exploring content originality of a social media post by amalgamating the features extracted from author's writing style and her online circadian rhythm. This originality detection approach is used to identify news dissemination patterns among news media community in Facebook and Twitter by observing news originators and news consumers. For this experiment, dataset is collected using our implemented crawlers through Facebook Graph API and Twitter streaming APIs. The next experiment is on exploring flaming events on news media in Twitter by using an improved sentiment classification model. The final experiment is focused on detecting topics that are discussed in a real-time meeting with the aim of generating a brief summary at the end.

The second contribution is to adapt Transfer Learning models for the clickbait detection task. We evaluated performances of three Transfer Learning models (BERT, XLNet and

RoBERTa), and delivered a set of architectural changes to optimize these models. We believed that these three models are the representatives of most of the other Transfer Learning models in terms of their architectural properties (Autoregressive model vs Autoencoding model) and training datasets. The experiments are conducted by introducing advanced fine-tuning approaches to each model such as layer pruning, attention pruning, weight pruning, model expansion and generalization. To the best of authors' knowledge, there have been an insignificant number of attempts to use Transfer Learning models on clickbait detection tasks and no any comparative analysis of multiple Transfer Learning models focused on the clickbait detection task.

Keywords

Clickbait, Transfer Learning, BERT, XLNet, RoBERTa, Deep Learnig, Sentiment Classification, Topic Detection, Originality Detection, Social Media, Facebook, Twitter, News Media

Résumé

Presque tous les internautes sont susceptibles d'être victimes de clickbait, suppposant à tort qu'il s'agit d'informations légitimes. Un type important de clickbait se présente sous la forme de spam et de publicités qui sont utilisés pour rediriger les utilisateurs vers des sites web. Un autre type de "clickbait" est conçu pour faire la une des journaux et rediriger les lecteurs vers leurs sites en ligne, mais ces nouvelles sensasionelles peuvent être trompeuses. Il est difficile de prédire le degré de click-baity d'une nouvelle donnée car les clickbait sont des messages très courts et écrits de manière souvent obscure. La principale caractéristique qui permet d'identifier les clickbait est d'explorer l'écart entre ce qui est attendu dans un post, le titre de l'information et l'information réellement présente dans l'article qui y est lié. Dans cette thèse, on propose deux approches innovantes pour explorer le clickbait généré par les médias d'information dans les médias sociaux. Les contributions 1) de proposer une approche multimodèle basée sur la fusion en incorporant des techniques d'apprentissage profond et d'exploration de texte et 2) d'adapter les modèles d'apprentissage par transfert (TL) pour étudier l'efficacité des transformateurs permettant de prédire le contenu des clickbaits.

Dans la première contribution, le modèle de fusion repose sur l'utilisation de trois caractéristiques principales, à savoir la similitude entre la publication et le titre, le sentiment de la publication et du titre, et la similitude d'actualité entre l'article de presse et la publication. Le modèle de fusion utilise trois algorithmes différents pour générer une sortie pour chaque caractéristique mentionnée ci-dessus et les fusionner à la sortie pour générer le classificateur final. En plus de la mise en œuvre du classificateur de fusion, nous avons mené quatre expériences étendues principalement axées sur les médias d'information dans les médias sociaux. La première expérience consiste à explorer l'originalité du contenu d'une publication sur les réseaux sociaux en fusionnant les caractéristiques extraites du style d'écriture de l'auteur et de son rythme circadien en ligne. Cette approche de détection de l'originalité est utilisée pour identifier les modèles de diffusion des nouvelles parmi la communauté des médias d'information sur Facebook et Twitter en observant les auteurs et les consommateurs de nouvelles. Pour cette expérience, l'ensemble de données est collecté à l'aide de nos robots d'exploration implémentés via l'API Facebook Graph et les API de streaming Twitter. La prochaine expérience consiste à explorer les événements enflammés sur les médias d'information sur Twitter en utilisant un modèle de classification des sentiments amélioré. L'expérience finale se concentre sur la détection de sujets discutés lors d'une réunion en temps réel dans le but de générer un bref résumé à la fin.

La deuxième contribution est d'adapter les modèles de Transfer Learning pour la tâche de détection de clickbait. Nous avons évalué les performances de trois modèles d'apprentissage par transfert (BERT, XLNet et RoBERTa) et fourni un ensemble de modifications architecturales pour optimiser ces modèles. Nous pensons que ces trois modèles sont les représentants de la plupart des autres modèles d'apprentissage par transfert en termes de propriétés architecturales (modèle autorégressif vs modèle d'autoencodage) et de jeux de

données de formation. Les expériences sont menées en introduisant des approches avancées de réglage fin pour chaque modèle telles que l'élagage des couches, l'élagage à l'attention, l'élagage au poids, l'expansion et la généralisation du modèle. À la connaissance des auteurs, il y a eu un nombre insignifiant de tentatives d'utilisation des modèles de transfert d'apprentissage sur les tâches de détection de clickbait et aucune analyse comparative de plusieurs modèles de transfert d'apprentissage axés sur la tâche de détection de clickbait.

Mots-clés

Clickbait, Apprentissage par transfert, BERT, XLNet, RoBERTa, l'apprentissage en profondeur, Classification des sentiments, Détection de sujets, Détection d'originalité, Médias sociaux, Facebook, Twitter, Médias d'information

Table of contents

Chapter 1

Introduction

Contents

1.1 Motivation

Social media platforms like Facebook, Twitter and Instagram allow anyone to publish or share their own thoughts and stories. These platforms provide space for non-journalists to reach a mass audience to engage in journalistic activities to produce journalistic outputs, including news [1]. As a result, vast number of news shared via social media platforms began to spread as news satire, news parody, advertising, propaganda, misinformation and disinformation. Hence, understanding a news item is real or misinformation is challenging especially in online social media platforms. Determining whether the news is fake or real depends upon the judgments of each individual such as checking the news source is credible, observing other reputable news sources for similar content, skim through the entire article instead of only reading the headline, inspecting facts like published date, examining whether the source is biased according to your own belief, and checking whether it is a known website that share satire stories. Some of these activities can be automated to recognize misinformation shared across social media platforms, but several facts cannot be automated to achieve considerable performances similar to human judgments. Investigations on fake news and misinformation began popular after 2016 USA presidential election as much of the news flooded through the Internet consisted of written pieces and recorded segments promoting false information or perpetuating conspiracy theories created with an intention to deceive for political gain. As a consequence, in this era, much attention has been drawn to explore misinformation shared across social media platforms in multiple ways.

News media often use different types of techniques to spread news items aiming to attract reader's attention. In general, these news media can be considered as reliable, but they are frequently driven by biases as many readers actively check for more entertaining and short form of news on social media such as visual storytelling with a small content. Therefore, news media habitually use propaganda-like motives and interesting headlines to deliver news to the audience, and sometimes they use deceptive headlines with the intention to attract a large audience. In this scenario, a reader can easily be a victim as she assumes the news source to be legitimate, but in reality, this 'news' can be deceptive, sensationalized, misleading, unverified and irresponsible information. These types of headlines or titles are known as Clickbaits that are in the form of false advertisements designed to attract readers' attention via thumbnail links leading to read, view or listen to the content available on their web page. Human can use some facts to judge whether a given headline is clickbait similar to identifying misinformation. However, automatically predicting the degree of clickbaityness need strenuous efforts due to different properties of the clickbait, in particular, written as a short headlines and requires proper semantic to understand with the knowledgeable facts.

The main goal of this thesis is to provide two approaches to detect clickbait in social media using recent Natural Language Processing methods such as Transfer Leaning and deep learning techniques. Through a content analysis of posts published by news media and their engagement outcomes in social media, we looked at the principle characteristics of clickbait content and their relative prevalence in order to explore clickbaits identifying features such as: i) similarity of the social media post and the news headline ii) sentiments of the social media post and the news headlines and iii) keywords/topics similarity between news headline, social media post and the news article. In addition, another approach we proposed in this thesis is to identify the contextual patterns of the clickbaits, in which we can train a model using manually labeled contents (clickbaits and non-clickbaits), and adapted recent Transfer Learning models to make the predictions. To this end, we use different datasets in multiple experiments including the data extracted from Facebook and Twitter using our implemented crawlers. Moreover, we also used external datasets shared by two different clickbait challenges to train and build novel models with the aim of detecting clickbait contents.

1.2 Objectives of the Thesis

In this section, we presents main objectives of my thesis in which each objective represents as one contribution. This thesis aims to provide two solutions to detect clickbaits in social media. The main objectives to achieve these aims are as follows:

- To propose a deep learning fusion-based approach to detect clickbait content in social media
- To apply Transfer Learning techniques to detect clickbaits and compare performance of these models with deep-learning based approaches

1.3 Contributions of the Thesis

Our approaches to achieve the above research objectives are organized into two parts as two contributions. We discuss each contribution as follows:

C1: The first contribution is on exploring a mechanism to detect clickbait posts in social media considering a set of features such as text similarity, sentiment value and topical similarity of texts. This contribution proposes a method to classify clickbaits using a combination of outputs received from each of those features, and finally, fuse them together at the output layer to make the classification. Main advantage of using fusion model for a clickbait classification task is the possibility of amalgamating multiple

content properties extracted from social media post, news headlines and news content. Our fusion model uses three features extracted from three different enhanced algorithms that are based on text similarity detection, topic detection and sentiment classification. In addition, we use these algorithms to evaluate set of different other applications and therefore, our first contribution provides solutions to four experiments as explained in the following sections. Finally, we deliver a novel methodology to detect clickbait content using deep neural networks by merging several content features. The datasets used in this contribution is mainly collected from Twitter and Facebook through our implemented crawlers and we also use two other datasets that are shared by clickbait detection competitions.

C1.1: The first experiment mainly uses the features extracted from similarity detection algorithms that are based on n-grams. In this experiment, we proposed a solution to identify content originators of social media posts based on their writing patterns and online circadian rhythm. Since each person has his/her own writing style, we can indicate this as a write-print which is similar to having a unique finger print. As a result, we assume that social media users use their own writing styles that lead them to have their own write-print. In addition, social media users have their own online behavioral properties (i.e. post sharing time patterns) which is named as their online circadian behavior. Since different users have thier own online circadian behaviour, this is another useful property we used in this experiment which is named as a time-print. Hence, in order to detect originators of an online social media post, we considered both write-print and time-print, and proposed a new framework called ConOrigina. We crawled news media data from Facebook and Twitter to explore news originators, news dissemination and news consumers based on our ConOrigina framework.

C1.2: The second experiment is also used text similarity detection approaches and the ConOrigina framework to understand news dissemination patterns in social media platforms. We used news media datasets collected from both Facebook and Twitter to analyze news self-originators (news media who originate content by themselves or publish fresh content), news providers (news media who distribute their content to other news media), and news consumers (news media who mostly share replicated content that have been al- ready published by other news media). In addition, we proposed a reader reaction predictive model for news media to increase news popularity based on the considered features that are explored from the ConOrigina framework.

C1.3: In the third experiment, we explored flaming events in social media using our

sentiment classification algorithm. Flamings occur when a social media post or a blog receive a flood of negative feedback, insults or offensive words. These comments usually posted to harass another person socially and psychologically. The detection and analysis of flaming event is possible with the use of sentiment values of each comment. Hence, in this experiment, we identified flaming events in news media on Facebook by utilizing the sentiment scores of the feedback received for posts.

C1.4: The fourth experiment amalgamates topic detection and segmentation algorithms in order to identify topics that are discussing in a real-time meeting, and then, use those topics to generate the meeting minutes. With few textual and structural features, detecting topics and providing a summary is a significant challenge. Segmenting the meeting transcript is the first step to detect topic boundaries. Topics in a text can be considered as a probabilistic clusters of words that are semantically related to each other and they are represented as a set of descriptive and collocated terms. Therefore, we apply topic modeling approaches to detect topic coherence in each segment. Based on those topics we build the topic shift and meeting minutes automatically.

C2: In the second contribution, our aim is to apply Transfer Learning models to detect clickbait content. There are insignificant number of previous works that have been using Transfer Learning for clickbait classification, but they directly applied models without any modification to the architectures or proper fine-tuning techniques. Hence, we tried to explore performance of Transfer Learning models on the clickbait classification downstream task. The main fine-tuning mechanisms considered in this work includes model pruning (both layer and weight pruning), model expansion and generalization. In addition, we used data augmentation strategies to generate a balance dataset that helps generating clear boundaries with respect to each label. We used two datasets in this contribution that are obtained from two different clickbait challenges.

1.4 Publications List

Journal Papers

- Praboda Rajapaksha, Reza Farahbakhsh, and Noel Crespi. "Scrutinizing news media cooperation in Facebook and Twitter." IEEE Access (2019).
- Praboda Rajapaksha, Reza Farahbakhsh, and Noel Crespi. "iTrip, a framework to enhance urban mobility by leveraging various data sources." Transportation research

procedia 24 (2017): 113-122.

Conference Papers

- Praboda Rajapaksha, Reza Farahbakhsh, Noel Crespi and Bruno Defude. "Uncovering Flaming Events on News Media in Social Media." 2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC). IEEE, 2019.

- Praboda Rajapaksha, Reza Farahbakhsh, Noel Crespi and Bruno Defude. "Inspecting interactions: Online news media synergies in social media." 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). IEEE, 2018.

- Praboda Rajapaksha, Reza Farahbakhsh, and Noel Crespi. "Identifying content originator in social networks." GLOBECOM 2017-2017 IEEE Global Communications Conference. IEEE, 2017.

Under Review

- Praboda Rajapaksha, Reza Farahbakhsh, and Noel Crespi. "Discern Clickbait from Non-Clickbait: A Transfer Learning Approach with BERT, XLNet and RoBERTa." ACM Transactions on Data Science.

Working Paper

- Praboda Rajapaksha, Reza Farahbakhsh, and Noel Crespi. "How does BERT Attention Pruning Affects on Clickbait Classification?." ACM Transactions on the Web (TWEB).

1.5 Relationship of Publications with Contributions

In this section, we provide the relationships of publications with contributions.

- The publications 'Scrutinizing news media cooperation in Facebook and Twitter', 'Identifying content originator in social networks', 'Inspecting interactions: Online news media synergies in social media' and 'Uncovering Flaming Events on News Media in Social Media' are correspond to Contributions C1, C1.1, C1.2 and C1.3 in Chapter 3.

- The submitted paper 'Discern Clickbait from Non-Clickbait: A Transfer Learning Approach with BERT, XLNet and RoBERTa' corresponds to C2 in Chapter 4.

1.6 Outline of the Thesis

The thesis is structured into five chapters.

- Chapter 1 describes the background of research topics, motivation, contributions of this thesis, summary of each chapter and the outline of my thesis.
- Chapter 2 presents the background and related technologies relevant to the main topics of this thesis, i.e., clickbait detection, multimodel learning, social media data collection etc.
- Chapter 3 presets the fusion-based clickbait detection methodology and other extended experiments conducted using implemented algorithms for sentence similarity detection, sentiment classification and topic detection.
- Chapter 4 presents how to optimize and use Transfer Learning models to classify clickbait through different fine-tuning strategies such as layer pruning, attention pruning, model expansion and data augmentation strategies.
- Chapter 5 summarizes the thesis and discusses possible future directions relevant to my research works.

Chapter 2

Background and Related Technologies

Contents

2.1 Overview

The background and related technologies presented in this chapter gives a general summary that are related to the main research topics of my thesis. Later on, a separate and detailed overview of the related work will be discussed for each study and experiment in this thesis.

2.2 State-of-the-art on Clickbait Detection

News content rapidly spread through social media and more often, misinformation attract many readers and spread faster than legitimate news, aided by humans and bots intentionally or unintentionally. The notoriety of fake news and misinformation can be partially attributed to clickbaits as well. A great deal of research has been conducted on clickbait detection in social media platforms focusing mainly the Twitter and Instagram [2] [3] [4] [5] [6]. Some other studies have been adapted data extracted from headlines of news articles, blogs, and other sources [7] [4] [8] [9]. Some works have focused only the textual content while others have used both text and images to characterize clickbaits [10] [11]. In addition, there are previous works that were aimed at detecting clickbaits in video contents [12] [13]. Researchers have used various NLP techniques including machine learning [5] [14] and linguistic differences or features [11] in order to detect clickbaits.

In recent times, clickbaits became very popular research topic and therefore, several competitions and challenges were announced on this regards. One successful competition was Webis clickbait challenge, introduced in 2017 [15], was designed to detect clickbait posts in Twitter. The competitors asked to develop a classifier to rate how click-baiting a social media post is. Their dataset consists of 19,538 labeled posts and 18,979 unlabeled posts [16] [17]. Competitor groups were ranked based on their models' mean square error (MSE) and 28 groups were participated to the challenge[1]. We observed that almost all the proposed approaches were based on machine learning and deep learning techniques.

Another competition was introduced by Kaggle [18] to explore various semi-supervised and Transfer Learning approaches to text classification. One usecase they considered was the classifications of articles into news, clickbait and other. Their dataset includes separate training (24,870 entries), testing (5,646 entries) and validation (3,551 entries) datasets. We observed from the leader-board that only 9 participants have participated[2] to the challenge, and they have mainly used Transfer Learning models without any additional improvement to the existing model.

Pujahari et al. [19] have proposed a hybrid classification technique to distinguish clickbait from non-clickbait by integrating different textual features, sentence structure, and

[1] https://www.tira.io/task/clickbait-detection/dataset/clickbait17-test-170720/
[2] https://www.kaggle.com/c/clickbait-news-detection/leaderboard

clustering techniques. During their preliminary categorization, headlines were separated using eleven features and categorized based on the syntactic similarity and finally, word vector similarity techniques have been applied for clustering. These categorized dataset is then used with machine learning models such as SVM, Decision Trees and Random Forest to perform the final classification. Rony et al. [8] have tried to detect clickbaits in social media using skip-gram model which helps to identify the similarity between two texts.

Video clickbait is also a critical problem in online video sharing platforms. Lanyu et al. [12] have proposed an approach to identify video clickbaits by exploring the comments received from the audience rather than analyzing video content itself. They mainly used semantic features of the comments such as sentiments and endorsements to reveal user's attitude and behavior towards a possible clickbait video. Zannettou et al. [13] have introduced deep learning based approach that detects clickbait video on YouTube using features extracted from headline, thumbnail, comments and video statistics. Omidvar et al. [5] have proposed another model using deep learning methods to find video clickbaits and they used bi-directional GRU for clickbait detection. In the previous research works, Transfer Learning models were applied to detect fake news [20] but, very little research works have been conducted on clickbait detection.

Language modeling and sentence representation learning is an important research area in various downstream tasks especially in the NLP domain. Previous studies have employed supervised models, unsupervised models and language based models such as ELMo (Embedding from Language Model) [21] (introduced in 2018). ELMo uses bidirectional Recurrent Neural Networks - RNN to improve word-embedding and shown better performance in many downstream tasks. With the disclosure of Transformer by Vaswani et al. [22] in 2017, various NLP models have been developed in the recent years. One of the most popular language modeling algorithm that use deep bidirectional transformers is BERT [23]. The initial use of BERT is to predict the next sentence, but it broke the records of previous state-of-the-art methods in eleven different NLP tasks. There have been many extensions to the BERT model such as RoBERTa, XLNet DistilBERT and ALBERT[3]. XLNet [24] is another transformer modeling approach that is designed to overcome the issues came up with the BERT model as it uses extra tokens that are not important during the training phase. Multitask learning and classification is an effective approach to share the knowledge extracted from several supervised tasks. Few recent studies have used BERT for multitask classification [25], [26] covering common text classification tasks such as sentiment analysis, question and topic classification. In addition, BERT has also been used for Irony detection in an Arabic Twitter dataset by Zhang et al. [27], Intention classification [28], and Fake news detection [29].

[3]https://github.com/huggingface/transformers

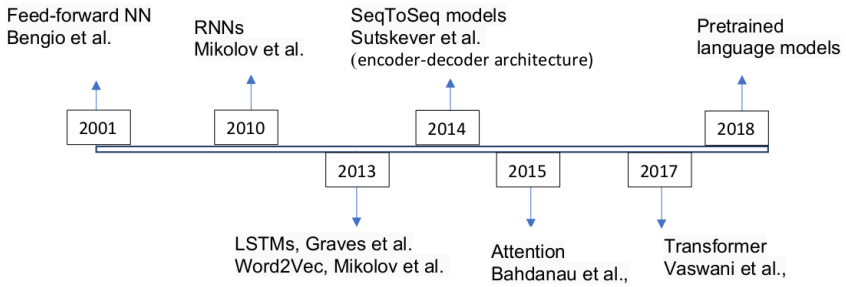


Figure 2.1 – Major advances in Natural Language Processing.

In the literature, no any work is proposed to build a deep learning model that can be automated to detect clickbait and therefore, as the first contribution of this thesis, our aim is to propose a deep learning fusion-based model which can amalgamate many textual properties to determine whether a given post is clickbait or not. As per authors' knowledge, a limited number of previous works have worked on clickbait detection using Transfer Learning. We aim to apply and evaluate Transfer Learning models to discern clickbaits from non-clickbaits by modifying and improving the architecture of the selected Transfer Learning models. We aim to modify the architecture of each model with different fine-tuning strategies to observe the best-performing model.

2.3 Major Advancements in NLP

Natural Language Processing-NLP, is a field of Artificial Intelligence, has started in nearly 1950s after Alan Turing published an article titled 'Computing Machinery and Intelligence' in which he introduced the Turing Test to identify the ability of a machine to exhibit intelligent behaviors similar to human. NLP is mainly concerned about the interactions between computer and human languages in areas such as speech recognition, natural language generation and natural language understanding. A brief overview of major advances to the NLP field is presented in Figure 2.1 which includes Feed-forward Neural Network, Recurrent Neural Network-RNN, Short-Term Memory-LSTM, Word2Vec, Sequence-to-Sequence models, Attention, Transformer, and pre-trained language modelling. More details about these advancements are explain in detail below.

With the growth of web in 2000, amount of raw language data became available and therefore, research has focused on supervised and semi-supervised learning models to conduct various research on understanding the insights of those data, but in this era, main trend is towards using Language modeling techniques to solve such research problems. Bengio et al. have proposed the first neural language model as a feed-forward neural network

in 2001 [30]. Later, Feed-forward neural networks are replaced with Recurrent Neural Networks-RNN and Long Short-Term Memory-LSTM [31]. Multitask learning, a method for sharing parameters between different models that are trained on multiple tasks, was first applied for NLP in 2008 by Collobert et al. [32]. They have used a shared look-up table between two models that were trained for two different tasks. In recent research works, multitask learning is used to share various patterns among models is becoming more important as these models are trying to improve their generalization capabilities. Even though word embedding and vector representation of words have been used in early 2001, the main innovation was proposed by Mikolov et al. in 2013 [33] [34] and they improved both quality of the vector and training speed by sampling more frequent words used to find phrases and relations in the text. After 2013, Word2Vec and word embedding have started to use for many applications such as named entity recognition-NAR (Lample et al. [35] in 2016) , language modeling (Kim et al. [36] in 2016) , etc. Most of these models uses CNN or LSTM that takes textual content as inputs and they outputs a character based text representation. Character-based representations such as n-grams shown better performances than considering individual characters for many tasks. Hence using word vectors with sub-words have become a popular research area in 2016 Wieting et al. [37], in 2017 Bojanowski et al. [38] and this was the base architecture used to develop fastText classifier introduced by Facebook in 2016 [39]. One main issue with word embedding was that it is mainly based on the pre-identified vocabulary words and therefore, cannot deal with words that have not been seen during training. Sub-wordings or considering n-grams was one of the easiest ways to mitigate this issue as introduced by fastText.

Neural Networks were introduced in 1990 by Elman et al [40] as a Vanilla RNNs and in 1997 Hochreiter et al. [41] proposed classic LSTM. However, around 2013 and 2014 neural network models began to apply in NLP tasks with the introduction to BiLSTM by Graves et al. in 2013 [42]. Although Convolution Neural Networks-CNNs were used for computer vision, CNNs were started to adapted with NLP task in 2014 [43]. CNNs and LSTMs can also be combined or stacked, and convolutions can also be used to speed up LSTM in which both models consider textual content as sequences. Another milestone in NLP is the discovery of sequence-to-sequence learning by Sutskever et al. [44] in 2014 which is based on neural networks aiming to map one sentence to another. Their model used encoder-decoder architecture and this was a major turning point in machine translation. In 2016, Google has replaced all its previous machine translation models with neural machine translation models [45]. This is the main model framework in today's language modeling tasks that are based on the encoder-decoder pair.

Attention introduced by Bahdanau et al. [46] in 2015 is another core discovery of neural machine translation in recent times. Attention was mainly introduced to overcome the

limitations of sequence-to-sequence model in which it always requires to compress the entire sentence into a fixed sized vectors. Attention alleviates this by allowing decoder to traversing back to the hidden state of the source sequences. Attentions are applicable for any tasks that need make decisions based on the input source. Most important point using Attention is that, weights are useful to explore which part of the input sequence is more important for the final decision making process.

Today's trend in NLP is to apply pre-trained language models. Firstly, a large corpus of un-annotated data is fed into the language model in order to learn how the language is written and other properties of the text. This model is then trained on a supervised task-specific dataset and fine-tune the model to achieve better results. As pre-trained language models only require unlabeled data, training can be scaled to new domains, new languages and huge number of tokens or parameters. Language models improved many state-of-the-art approaches in a significant way. Since, pre-trained language models need less data to train for a specific task, these are applicable for any low-resource languages that are having less number of labeled data.

As shown in Figure 2.1, many advancements to the NLP are introduced after 2010. My thesis has started in early 2017 and therefore, the proposed clickbait detection methods and related algorithms were mainly based on n-gram, RNN, Word2Vec and deep learning models that were proposed and applied to NLP in that period as a multimodel fusion technique. However, with the emergence of pre-trained Transfer learning models for NLP in 2018, I have started to apply those techniques to the clickbait classification task and achieved significant improvements over the traditional methods. Literature review on the traditional NLP methods related to multimodel learning and a comprehensive review on Transfer Learning is provided in the following sections.

2.4 Multimodel Learning

In recent years, the term big data has become an interesting topic which is able to recognize trends, patterns and complex associations related to human interactions and behaviors. One main challenge when using big data for different experiments is that these data are multivariate in different types, formats (structured, unstructured, semi-structured), and they are extracted from different sources having multiple properties and categories. The data extracted from different modalities (modality refers to the way in which something happens or is experienced), such as linguistic, visual messages and acoustic signals can be integrated together to address one of the original goals of Artificial Intelligence (AI) which is named as multimodel machine learning. Some examples of modality includes natural language (both written and spoken), visual (image and video), auditory (voice, sound and

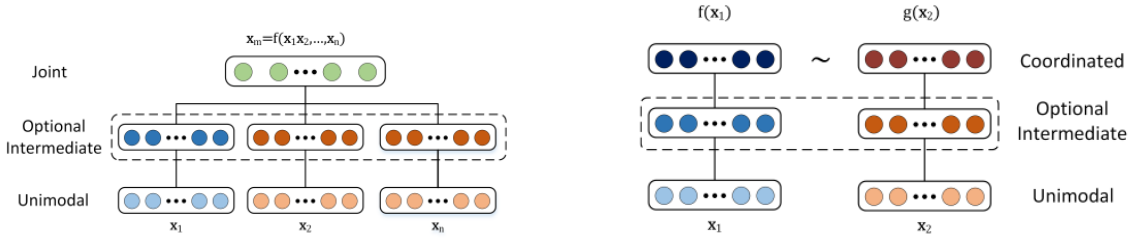
music), touch, smell, infrared images etc. Multimodel machine learning is very challenging due to heterogeneous data.

Multimodel research were conducted in the early years since 1970 to identify behavioral patterns, to perform computational aspects on audio-visual speech recognition, human computer interactions, multimedia computing, human informational retrieval and etc. In the following era deep learning came in to the research since 2010 starting with the representative learning. Some examples include multi-modal deep learning [47], multi-modal learning with deep Boltzmann machines [48] and Visual attention [49]. Deep learning based multimodel research works have been successful due to the availability of large-scale multi-model datasets, existence of GPUs and dimensional linguistic features.

There exists five core technical challenges related to multimodel learning: representation, translation, alignment, fusion and co-learning [50]. Representation is the first challenge that needs to identify how to represent and summarize heterogeneous multimodel data (e.g. language is often symbolic while video and audio are signals). Next, the second challenge is the translation where we need to map data obtain from one modality to another (e.g. identifying a relationship between image representation and textual content representation). The third challenge is the alignment in which we need to identify different relationships in between different modalities. Joining data from multiple modalities is known as fusion that is considered as the fourth challenge in multimodel machine learning. The fifth challenge is co-learning which explores how to transfer knowledge between modalities [50]. In this thesis our aim is to use fusion multimodel learning approach as we can amalgamate many features obtained from different representations to detect clickbait content.

2.4.1 Multimodel Representation Learning

One major machine learning challenge is to represent data in such a way that any computational model can understand. Representation learning is introduced to make much easier to extract useful information from datasets that can be used to build predictors and classifiers [51]. The representation of multimodel data poses many difficulties such as merging heterogeneous data together, dealing with different levels of noises and also dealing with missing data. The performance of machine learning model depends on type of the representation. There are some evidences to prove this concept in speech recognition [52] and visual object classification [53]. Many multimodel representations includes unimodel representation [54]. Unimodel representations were extensively introduced for images using neural architectures such as Convolutional Neural Networks (CNN) [53], speech recognition [52] and in Natural Language Processing as word embedding [55]. There are two types of representations that can be identified in multimodel representation learning, namely, joint representation and coordinated representation where their architectures are shown in Fig-



(a) Joint representation

(b) Coordinated representations.

Figure 2.2 – Structure of joint and coordinated representations in multimodel representation.

ure 2.2. As illustrated in Figure 2.2a, joint representations are usually project all unimodel representations in to the same space. On the other hand, as illustrated in Figure 2.2b, coordinated representation use their own space instead of projecting into a single space. One example for a joint representation is concatenation of individual modality features which is referred to as early fusion [54].

Neural Networks can be considered as one of the most popular methods for a unimodel representation that are widely used to represent textual data and visual data, and can be used with multimodel representations as well [56], [57]. When neural networks are applied on multimodel representations, each modality starts with a separate neural layer followed by a hidden layer, and then projects hidden layer to a single joint space. The joint multimodel representation passe through many other hidden layers before returning prediction results. These neural networks can be trained using an autoencoder on unsupervised data [56], in which auto-encoders can be used to represent each modality individually and fuse together at the end to generate multimodel representations. Multimodel auto-encoders are also used in the previous research works [58]. The advantage of using neural networks for multimodel representation is that it gives higher performances and the model can be trained as an unsupervised model. Lack of direct solution to handle missing data is the main challenge in the neural network models [56]. Recurrent Neural Networks (RNNs) is another unimodel representation which is capable of dealing with the features extracted from different lengths of sequences of words, audio and video [59]. RNNs are also used in the multimodel representation in the literature [60]. Another representation category is coordinated representation which uses unimodels and these unimodels learn separately by coordinating among them. One example is similarity models that minimize the distance between each modality in the coordinating space. For example, an image of a dog written 'dog' in it have a smaller distance in the coordinating space than the same image with the word 'tree' [61]. Neural networks have also used in the coordinated representation as they can learn jointly coordinated representation throughout the process [62]. With the above facts, we believe that

unimodel representation with RNNs give better performances when using many features from different representations that are obtained from text, image, video and audio.

2.4.2 Multimodel Fusion

Multimodel fusion is a concept derived when merging multiple modalities in order to do some predictions such as classification and regression. This concept allows to do the predictions on one particular task using many modalities which further improves the accuracy and precision. In addition, multimodel fusion is useful when an outcome of one modality is not available while predictions from other modalities are convenient to obtain final results.

Many multimodel fusion methods have followed model-agnostic approaches [63] where each model can be split into early (feature based), late (decision-based) and hybrid fusion [64]. One important aspect of the hybrid fusion is that it can merge outputs obtained from each individual modalities when they are executing as early fusion models [65]. Late fusion uses individual decision values obtained from all unimodels and fuse them at the end by applying fusion mechanisms such as learned model [66], averaging [67] or voting [68]. However, late fusion ignores the low level of interactions among different modalities, instead it is directly works with generated outputs from multiple modalities. Even though model-agnostic approaches are easy to implement as machine learning models, they were not designed to support for multimodel data. Hence, there are some specific approaches that have been designed to perform multimodel fusion namely, neural networks, graphic models and kernel-based models. In this thesis our main focus is on the neural network based fusion models.

In the literature, neural networks have been applied to fuse information on different applications such as gesture recognition [69], video description generation [70], media question and answering systems [71] etc. In my thesis, the main idea is to fuse information by merging hidden layers of neural networks even though the architecture, used techniques and optimization techniques differ from each proposed methods. Multimodel fusion neural networks have been adapted RNNs to perform audio-visual emotion classification [72], continues emotion prediction [73] etc. The main advantage of using deep neural networks in fusion models is they can be trained using a large amount of data corpus and allow training both fusion and multimodel representation modalities throughout the process. In compared with non-neural network model, these models exhibit high performances in the literature with accurate predictions. However, in these models it is very difficult to identify which modality is directly relied on the final prediction and which features are more important.

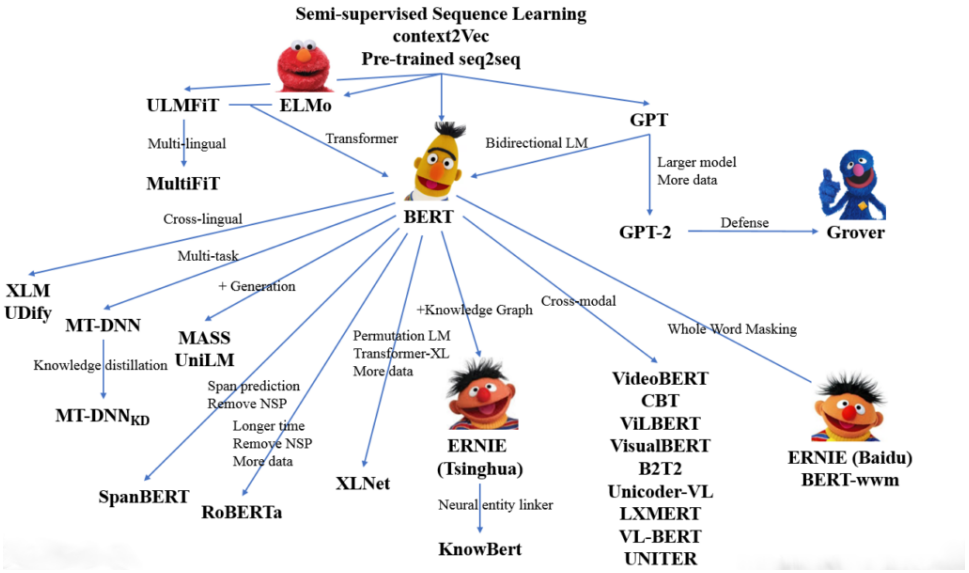


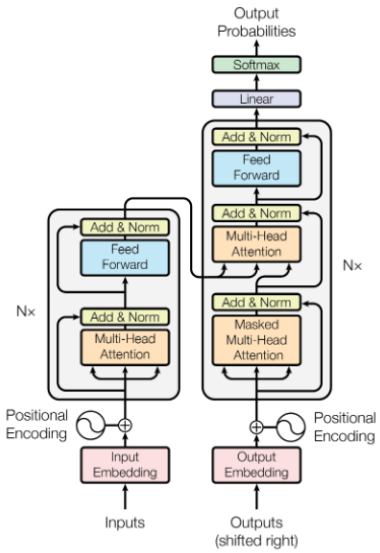
Figure 2.3 – Evolution of pre-trained Transfer Learning models.

2.5 Transfer Learning

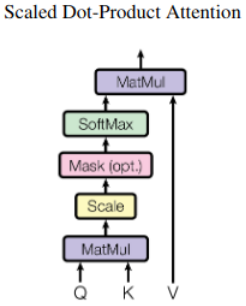
Transfer Learning is a technique in which a model developed for one task can be used for another task enabling rapid development progress and achieving improved mode performance even without a large training dataset. As show in Figure 2.3, the base model of Transfer Learning can be identified as the Sequence-to-sequence(Seq2Seq) architecture, a neural network architecture which transforms a given sequence of elements into another sequence. A popular example for a Seq2Seq model is Long-Short-Term-memory (LSTM) where it keeps track of sequence important data. Seq2Seq models usually consists of Encoder and Decode (Encoder maps input sequence to a high dimensional vector and Decoder turns the abstract representation of vector to the output). A single LSTM layer can be the basic form of the Encoder and Decoder. Attention mechanisms evaluate the sequence and decide other important parts of the sentence which is similar to identifying and keep recording important keywords or context. Hence, for each input, decoder decides what are the important keywords in a sentence and assign relative weights to those keywords. Even though RNN and LSTM are capable of having memories about the previous words and context, when a longer sequence is provided to these models they make references only to the very first few words in the sentence and references to the remaining words are not guaranteed due to lack of memory resources as they consist only short term memory. On the other hand, Transformers are capable of relating to the position of word in a given sentence in any length and sufficient resources are available to process data.

The paper 'Attention Is All You Need' done by Vaswani et al. [22] proposed a novel architecture named as Transformers that uses Attention mechanism explained above. Transformer is capable of transforming one sequence to another using Encoder and Decoder architecture, but it differs from previous Seq2Seq architectures. Figure 2.4a clearly shows the architecture of the Transformer, in which Encoder is at left side and Decoder is at the right side. We can observe it consists of different modules having many layers including Multi-head Attentions and Feed-forward layers. Transformer inputs are first decoded into n-dimensional space (input embedding and output embedding in Figure 2.4a) aiming to map words into feature space (by referring to a lookup table) in order to process and understand features by the computer. Therefore, at the initial stage input tokens are embedded with the embedding vector. In this architecture, there is no any LSTM memory modules to keep record on the context of the sentences and therefore, model has to learn positions of each word in the input sentence as a relative positions. These positions will be added to the n-dimensional vector space as embedding. Hence, in order to aware on word positions of the input sentence it uses Positional Encoding by adding an additional embedding vector of the same dimension to each word. The positional embedding is generated by assigning Sin and Cosine function for each word, i.e., for every odd time-step use a Cosine function and use a Sin function for every even time-step, and finally add those functions together with the corresponding embedding vector to generate network information on positions for each vector.

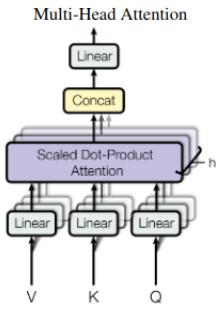
As shown in Figure 2.4b Attention uses three fully connected layers, namely query-Q, key-K and value-V. Q is a matrix containing a set of queries that are used to define weights for each word in the sequence influenced by all the other words (represented by K). SoftMax function is applied to each weight to have a distribution among 0 and 1. Figure 2.4c shows how these Attentions can be parallelized as linear projections by multiplying Q, K and V with the weight matrices that are learned during the training. Q, K and V vectors split into N number of vectors before applying self-attention, which is named as a head. Each head separately generate an output and concatenate together to make a single vector before passing through the final linear layer. Add and Norm functions shown in Figure 2.4a) are mainly used for layer normalization. The output from multi-head attention is added to the original positional embedding and then pass through feed-forward network which comprised of a set of linear layers activated with ReLU. Next, a point-wise feed-forward layer is used to project the attention outputs for further processing that allow to have a richer representation of the output vectors. This output goes though a linear layer acting as a linear classifier and then the resulting vector passes through a Softmax layer in order to produce probabilities in between 0 and 1 . Index of the highest probability is equal to the predicted classification class. This is the mechanism of a Transformer that can be used



(a) The architecture of the Transformer.



(b) Scaled Dot-Product Attention.



(c) Multi-Head Attention consists of several attention layers running in parallel

Figure 2.4 – The Transformer architecture and Attention mechanism.

to make better predictions with the Attentions.

In today's research activities, all pre-trained models use Transformers to perform different tasks on texts such as classification, information extraction, question answering and summarization in multiple languages. Hugging Face library [4] provides a set of API calls to access Transformer libraries in Python. This library uses PyTorch and ported to TensorFlow 2.0. Hugging Face library uses three main classes:
the configuration class indicates related information about the model such as number of layers, hidden size etc,
the tokenization class converts strings to arrays or tensors of integers and different TL models uses their own tokenisation class, and
the model class uses to configure rules on the neural network modeling aspects.

Hugging Face library provides API calls to fine-tuning a Transformer model with less efforts as it provides the accessibility to different inbuilt functions such as load data, preprocess data, hyper-parameters declaration and model training. We use the Hugging face library in our implementation to fine-tune different Transfer Learning models with various configuration changes. Hence, we do not use Keras library in our implementations and all the changes are done on the Transformer class itself.

[4] https://github.com/huggingface

2.5.1 Transfer Learning (TL) models

To date, there are 33 Transformer models available in the Hugging Face library [5]. Each model in the library can be either Auto-regressive model, Auto-encoding model, Sequence-to-sequence model, Multimodal model or Retrieval-based model.

Auto-regressive models are based only on the Transformer Decoder component, and use an Attention mask to identify the tokens that are already seen and to restrict accessing future tokens. The widely used application of the Auto-regressive models is text generation. Example models are GPT, GPT2, XLNet, Reformer and Transformer-XL.

Auto-encoding models mask and corrupt a set of tokens identified from the input sentence and then, try to predict the original input sentence. They use only the Transformer Encoder component of the model. These models can access the entire input sentence and build as a bidirectional representation of the whole sentence. Examples for Auto-encoding models are BERT, ALBERT, RoBERTa, DistilBERT, XLM etc. The main difference between Auto-regressive and Auto-encoding models is about the way how pre-training perform.

Sequence-to-sequence models use both Encoder and Decoder from the original Transformer and these models can be used applied on sentence translation, question answering and summarization tasks. The original Transformer model is one such example of these types of models and T5 is the latest model proposed by Google AI.

Multimodal models are capable of merging different features obtained from many input categories such as text, images and videos can be integrated together targeting a particular task. MMBT (Supervised Multimodal transformers for Classifying Images and Text) model is one such model. Retrieval-based models use some techniques identified from document retrieval that is applied during pre-training mainly for question answering systems.

Choosing which model is suitable for which task is a complicated process. However, based on the previous research works Pratik [74] have explored and proposed a diagram on what models are performing better for different tasks. They categorized NLP tasks such as classification (English, non-English, multi-language and long text), translation, knowledge graph, semantic search, question answering system, conversational agents and summarization. For each of these NLP tasks he explored models that performed better in the literature and this helps to understand about when we are choosing a model for a particular NLP task.

In this thesis we use both Auto-regressive and Auto-encoding models to evaluate best performed model for clickbait classification task.

[5] https://huggingface.co/transformers/model_summary.html

2.5.2 Exponentially Bigger Transfer Learning Models

Neural networks are over-parameterized. Reducing the size of a model is important during the computation as environmental cost is exponentially increasing [75]. There are three different ways that we can use to reduce the size of a model. They are Distillation, Pruning and Quantization.

Distillation DistilBert is 40% smaller and 60% faster than the other existing Transfer Learning models [76]. This model uses pre-trained BERT as a teacher model and generated as a student model by reproducing generalization capabilities of the teacher [77]. For instance, BERT is trained to predict [MASK] tokens and therefore, DIstilBert is also generalized into the similar approach. In the knowledge distillation, we calculate cross entropy between output of the student and output of the teacher. This model is smaller, faster and light weight and used a triple-loss mechanism by combining language modeling, distillation and cosine-distance losses. In DistilBert, many operations used in the Transformer architecture are highly optimized including the linear layer and layer normalization process. The analyses have shown that, variations on the last tensor or the hidden layers have no significant impact on the efficiency of the model, but changing the number of layers in the model impacts on its performance.

There exists many distillation based models that are proposed very recently. MiniBERT [78] is one model which tries to keep only 3 layers out of the original BERT model, and therefore has fewer embedding parameters and fewer hidden units. MiniBERT uses distillation and trains student the from scratch using logits on unlabeled data and then, fine-tune with the labeled data that were used to train the teacher. This has been mainly proposed for multilingual applications. In another study [79], researchers have proposed an accurate model to fit on a given memory and latency. They have considered different aspects including the parametric form (architecture, number of parameters, trade-off between number of hidden layers and embedding size) and the training data (size, distribution, presence or absence of labels and training objective). They have used only 6 layers in the student model and trained with BERT 12 layer teacher model. Another study done by Tang et al. [80] have considered student model as a BiLSTM with a non-linear classifier. However, they avoided any additional tasks such as attention and layer normalization. TinyBERT [81] is another Transformer distillation model that can be trained for both the pre-training and task-specific learning.

Pruning Pruning directly works on the teacher model and remove weights from the teacher model to make it smaller. There are various ways we can prune and one simple way is to remove the attention heads of Transformers. Voita et al. [82] have observed that most important and confident heads plays a major role in the pre-training phase. When pruning the model, these specialized heads can be identified and prune the remaining heads that help

models to achieve its performances without degrading the original performances. Each head can have different roles such as positional heads(attending to an adjacent token), syntactic heads(attending to tokens in a specific syntactic dependency relation) and attention to rare words (heads pointing to the least frequent tokens in the sentence). Their pruning method is based on the continuous learning scenario starting from the full model and then identify the role of each head that are remaining in the model. Michel et al. [83] have identified that, considerable number of heads can be removed during testing and therefore, some layers in the model can be reduced to a single attention head. They proposed an algorithm that greedily and iteratively prunes away attention heads and their approach has resulted in up to a 17.5% increase in the inference speed on the BERT-based model. Weight pruning is another pruning method which is a technique that used to remove weights Transformer. Wang et al. [84] have introduced a novel structured pruning technique based on factorization and regularization techniques. Their results have demonstrated that, pruning a large model yields much higher performance than training a small model from the scratch. Layer pruning is another pruning technique which targets on removing full layers from the Transformers. Fan et al. [85] proposed this method and they have removed random layers during training, similar to dropouts, and then model learns to behave without those weights at the end.

Quantization The quantized models in PyTorch converts a float model to a quantized model with static 8 or float 16 data types for the weights and dynamic quantization in order to reduce the size of the model.

2.5.3 Transfer Learning Models Require Exponentially More Data

In general, NLP models depends their training and fine-tuning on more data. When two Transfer Learning models compared in terms of the datasets used for training, it is hard to say which model performs better. Because, performance of the model can be improved based on more data or performance can be enhanced due to the novel architecture of the model. One example is XLNet [24], an Auto-regressive model, which uses random permutations to learn the context of both sides. XLNet is trained on a large dataset than BERT (10x more data compared to BERT) and its architecture is different to BERT (BERT is an Auto-encoding model). In order to measure which model performs well, researchers have tried to use the same dataset for both BERT and XLNet and they also have ensured that almost every other parameters are similar. Trained on the same data with an almost identical training properties, XLNet outperforms BERT by a considerable margin on all the datasets. XLNet trained on more data under-performs the model trained on less data on some of the benchmark solutions [86].

Pre-training on more data Pre-training on more data does not guaranteed that the performance of the model will be high. The model performance is mainly based on the

number of model parameters, size of the dataset and the amount of computational power that is necessary for training the model. Model depth and width are not the parameters to define the performance [87]. Alon Talmor et al. [88] have conducted one study to understand whether the performance of language models need to be attributed to the pre-trained representation or to the process of fine-tuning? They used BERT and RoBERTa models as they have similar architectures, but BERT is trained on 135 billion tokens while RoBERTa is trained on 2.2 trillion tokens. They found that, different models with identical structure and objective functions differ both quantitatively and qualitatively. However, RoBERTa reasoning better and learn more common sense (e.g. age of a person, birth year) than BERT just only by analyzing their pre-trained models.

Fine-tuning on more data Fine-tuning is the process of using pre-trained models and adapting them to a particular task. Recent datasets are not too big and therefore easy to solve with little generalization and abstraction functionalities. For example, if there are two models (model A and model B), and model A trained with 100 training samples (does not improve its performances with more data) and achieved 90% accuracy while model B is trained with one million samples and achieved 92% accuracy [89]. In this scenario, model A performs better than model B due to its sample efficiency. This implies that the model performs better after adding one sample. Another factor is that, instead of training a large model we can have a very similar model that work well for fine-tuning. Online code length can be used to evaluate these models and we can get an idea about how much efficiency we can achieve by adding additional sample data [89]. The online coding is rooted in information theory and it is based on connections between generalization, compression, and comprehension.

2.5.4 In-domain vs out-of-domain Generalization

After training a Transfer Learning model with an existing dataset, final aim is to apply these models in to a real-world application. The transfer of training or generalization is the process which evaluates how learned parameters will be repeated in a new situation. In general, we mainly perform in-domain generalization, but it is important to reuse previously acquired knowledge about linguistic, lexicon and other features to new tasks quickly. When there are many datasets available for a specific problem it is better to generalize the model in order to achieve high accuracy. Usually, models train on large unlabeled corpus performs well in many different domains and the models can then be used for various other NLP tasks by adding an extra task-specific final layer after fine tuning on a supervised dataset for the task of interest. Another approach we can use in generalization is that train models on multiple task simultaneously. For example, question and answering model can be trained on other NLP tasks by formulating every task as question answering problem in order to

make the input and output of the model be consistent.

McCoy et al [90] have studied on improvement of the generalization of a model when it trains multiple times on the same dataset mainly by fine-tuning with different random seeds (difference is very small). They used BERT with Multi-genre Natural Language Inference (MNLI) dataset and evaluated them on the HANS dataset, which measures syntactic generalization in natural language inference. They obtained remarkably similar efficiency for all the runs of MNLI dataset with BERT, but some models varied widely in their generalization problem and they observed that the accuracy of the generalized dataset is varied with different trained models. This implies that, in-domain test performance gives no indication of how the model will behave in the real-world, and for some models it is hard to specify how the model will behave in real-world task unless it is applied to the. Another approach is to generate a new dataset instead of using heuristics as the study done by McCoy et al [90]. For example, a new sentence can be built by combining varies parts of the sentences and this is useful for generalization with limited data [91]. Hupkes et al. [92] have also tried to generate new dataset considering the repeating patterns of the sentences.

Radford et al. [93] demonstrated language models can perform down-stream tasks in a zero-shot setting without any parameter or architectural modifications. They have demonstrated that in a zero-shot setting, language models can perform wide range of tasks. BART [94] is a model proposed by Facebook AI that uses pre-training as a text-to-text objective. BART is mainly trained with corrupted texts and allow the model to reconstruct the original sentence. This model can be considered as the generalization of both BERT and GPT. Google has proposed Google T5 (Text-to-Text Transfer Transformer) [95] which is introduced as a framework that uses TL techniques for NLP in order to convert every language problem into a text-to-text format. This model is pre-trained on de-noising texts and fine-tuned in text-to-text format in which, model takes text as an input and produce text as output. This T5 framework can be directly to any task using the same model, objective, training procedure, and decoding process. With this approach, we can compare effectiveness of different Transfer Learning techniques with different datasets. The main advantage of this model is that we do not need to add any additional layers for fine-tuning, and we can use the same architecture for pre-training and fine-tuning and hence, no need to pre-train or fine-tune from the scratch (zero-shot learning).

2.5.5 Robestness of Transfer Learning Models

Recent works shown for evidences that fine-tuned Transfer Learning models for specific tasks can significantly shows better performances that could be achieved through training a model with the target tasks' data. This means that fine-tuning a large model with a thin layer output for a given task can achieve good results better than a carefully designed

task specific model without such pre-training [96]. This approach is really useful when only small amount of fine-tuning data are available for the target task. The model need to learn through the new target tasks and able to solve different input distribution and output label space than was seen in the pre-training phase, and also need to avoid over-fitting. Therefore, it is important to improve both robustness and effectiveness of the target task fine-tuning model. At first, the model can trained on unlabelled data and then model is further trained on an intermediate level using labelled-data. Finally, it is fine-tuned further on the target task and evaluate [96]. Another technique is to do the regularization in the process of creating dropouts from the weights and inputs in the middle layer of the Transfer Learning model. Instead of setting weights to 0, we can replace weights by the pre-trained value. Mixout, the method proposed by Lee et al. [97] stochastically mixes the parameters of two models and regularizes learning to minimize the deviation from one of the two models and that the strength of regularization adapts along the optimization trajectory. Since the limited data at the fine-tuning phase leads to over-fitting, Jiang et al. [98] proposed a new learning framework for robust and efficient fine-tuning for pre-trained models to attain better generalization performance. Finally we can train models on multiple tasks that allows to gather lots of data together and then train several model, and ensemble them. In a case where the model is too big, we can distil them as proposed by Microsoft [99], but this is very complex task.

In general, Transfer Learning models are brittle and spurious. Brittle means the model behave unexpectedly when the text is modified and Spurious means the model get its best performance with the least amount of work that memorizes artifacts and biases instead of truly learning, which indicates that our models are fragile. In both these cases, when we give a dataset with small changes the model fail in an unexpected way and gives wrong predictions. To avoid this problem, we should care about the linguistic patterns of the dataset. These linguistic patters are very useful to generate the validation dataset in order to evaluate the failures if we provide good inductive biases. The compositionality is also important to have a good understanding of the meaning of the sentence and therefore, it helps to incorporate some linguistics to the models. Models cannot learn common sense or facts about certain entities and certain types of information from the raw text. Hence, we need to train the model with these information by feeding them externally with the use of a knowledge base, visual representations or human in the loop (e.g. dialogue). In order to build datasets and analyse common senses, there are few common sense challenges introduced recently: Winogrande (AAAI 2020), Physical IQA (AAAI 2020), Social IQA (EMNLP 2019), Cosmos QA (EMNLP 2019), VCR: Visual Common sense Reasoning (CVPR 2019), Abductive Commonsense Reasoning (ICLR 2020) etc.

2.5.6 Data Augmentation

Dealing with an imbalanced dataset is a common challenge when solving a classification task in NLP. Data augmentation is one strategy that can be considered to make unbalanced dataset to make a balanced dataset among the number of classes. SMOTE [100] is a popular data augmentation strategy which uses a k-Nearest Neighbors classifier to create synthetic data points. A general downside of this approach is that synthetic examples are created without considering the majority class resulting in ambiguous examples.

There are 2 possible approaches to create dataset using SMOTE. In the first approach, we can apply SMOTE to the entire dataset and then split the dataset into training and validation. This approach is possibly over-estimate its accuracy and other performance measures since the validation dataset consists of similar instances. In the next approach, at first the dataset is divided into training and validation, and then apply SMOTE only to the training dataset. In this thesis, I use the later approach to make the dataset balance across all the labels.

2.6 Data Collection Strategies

This section explains about data collection approaches and data storage mechanisms used in my thesis for different analyses and how they affects with General Data Protection Regulation (GDPR).

2.6.0.1 Twitter Crawler

One of the main data sources used in my thesis is extracted from Twitter and it is mainly used for the contribution C1. Twitter data can be collected using Twitter streaming API which makes an HTTP request to get data from endpoints [6]. Results returned by the standard Twitter API are in JSON format. On June 19, 2019, Twitter began limiting access to the standard, statuses, mentions timeline, status and user timeline endpoints to 100,000 requests per day by default. Companies that manufacture products for other projects need to enter into a trade agreement to continue to access data at high levels. The standard API rate limiting is primarily done per user or per access token. The rate limits are divided into 15 minute intervals and there are two initial compartments available for GET requests: 15 calls every 15 minutes, and 180 calls every 15 minutes. In our implementation, we use the latter. With standard Twitter API, we can collect only the most recent 3,200 tweets in a given time frame. Premium and Enterprise APIs are subscription services while the Standard API is free and limited by the number of requests that can be made in a given

[6] https://developer.twitter.com/en/docs/tutorials/consuming-streaming-data

time interval and Premium/Enterprise API is primarily limited by the amount consumer is willing to pay. Since our aim is to use Twitter dataset for research purposes, we use Standard API service to access endpoints.

There are different ways to crawl data from Twitter Standard API, i.e. accessing user timeline and collecting data specifically on a hashtag. We implemented both crawlers, but our main analyses focus only on the data extracted from user timelines where we specify Twitter username to extract data from its timeline.

2.6.0.2 Facebook Crawler

The Graph API is the primary way for apps to read and write to the Facebook social graph. Social graph consists of nodes (user, photo, page, comment), edges (connection between collection of objects) and fields (data about an object i.e. page name, birthday etc). Through Graph API we can use nodes to get data about objects, edges to get collection of objects on a single object and field to get data about single object in a collection [7]. Graph API uses HTTP request/response to get/send data and every request requires an access token which can be generated by implementing an app to access Facebook login. Almost all requests are passed to the graph.facebook.com host URL and in order to access fields, nodes and edges we can specify those parameters with the host URL. For some requests we need to have an access token to access particular data from some users such as to download a photo and to extract posted text content etc. Our Facebook crawler is implemented using Facebook Graph API and we mainly used it to access data from user timeline to extract content that they are posting in their timeline.

2.6.0.3 Data Storage Mechanisms

Twitter and Facebook API return responses as JSON objects. Hence, we store every response in a MongoDB [8] database as a JSON object without adding any modification to the response. MongoDB can store JSON-like documents that supports arrays and nested objects. Also; it supports for powerful query language that allows to filter and extract any field regardless of how nested the it is stored in the document. Hence, all responses retrieved from our crawlers are stored in a MongoDB server for ease of use.

2.6.0.4 Facebook and Twitter Dataset, and GDPR Compliances

The General Data Protection Regulation (GDPR) [9] is a regulation in EU law on data protection and privacy in the European Union (EU) and the European Economic Area

[7] https://developers.facebook.com/docs/graph-api/overview/
[8] www.mongodb.com/
[9] https://gdpr-info.eu/

(EEA). GDPR addresses how personal data allows to transfer outside EU and EEA and give control to individuals over their personal data records in online platform. Organisations that process individual data have to compliance with the new legislations of GDPR and in case of academic research, the organization can be consider as the university. Data controllers must specify data collection and data processing procedure according to the law and must state how long data is being collected, retained and whether these data being shared by any other third parties or outside of the EEA. The regulation also applies to organisations based outside the EU if they collect or process personal data of individuals located inside the EU. The regulation does not apply to the processing of data by a person for a purely personal or household activity and thus with no connection to a professional or commercial activity. In addition, this regulations does not apply to the processing of personal data for national security activities. Data owners or data subjects have the right to request any dataset collected from them through a data collector and also, they have the right to request to erase their data under some constraints. A user must be able to transfer her personal data from one computer to another that has been collected by a data controller. The regulations does not apply to a dataset that has been sufficiently anonymized, but if it apply for is partially anonymized datasets enabling to identify individuals with remaining possible links.

GDPR was not designed to impede research activities and it recognized that any data can be useful for conducting research on certain topics. Hence, research can be exempt from the data processing and storage limitation unless other data protection principles met and specify accordingly. Universities as public authorities can state most likely lawful basis as 'task in the public interest'. When processing sensitive categories of data such as personal data about health, political opinion, religious believes, ethnicity etc, data controllers need to specify that this data is necessary for achieving purposes in public interest, scientific and statistical purposes.

The GDPR was adopted on 14 April 2016, and became enforceable beginning 25 May 2018. At the time of crawling data from our Facebook and Twitter crawler, GDPR was not adapted and therefore, we collected and stored social media data of news media presence in Facebook and Twitter and also collected user feedback received on certain public posts. All the information collected from our crawlers were content that were publicly available and no any private or personal records. After crawling and storing data in MongoDB, we made user feedback data anonymized as the main purpose is to use only the textual content and timestamp in our analyses. Hence, all data records use in our experiments are GDPR compliant.

Chapter 3

Clickbait Detection Using a Fusion Model

Contents

3.1 Introduction

The main focus of this chapter is to propose a new framework to detect clickbaits in social media using a fusion model which combines multiple features such as text similarity, sentiment value of a text and topics in a text. Our clickbait classifier uses a combination of outputs received from the algorithms that are designed to create above mentioned features and combine together as a fusion model. Using multiple textual features to detect clickbait content is helpful since we can amalgamate many properties extracted from the social media post, news headline and news content. In addition to proposing a clickbait classifier, we also provides different solutions to a set of other experiments that are mainly focusing on news media in social media. First, we explain different algorithms implemented with the related experiment and then, describe how those algorithms have been applied to detect clickbaits in social media followed by the challenges occurred during this research work.

3.2 Extended Experiments

As explained in the previous section, we implemented and used a set of algorithms for clickbait detection task. We conducted 4 experiments by adapting those algorithms in different usecases. Following explain these experiments, results and observations in details.

3.2.1 Experiment 1: Content Originality Detection in Social Media

In the first experiment, we used text similarity detection methods to identify possible content originators. Our aim is to classify authorship of a social media post based on the user's writing style and online circadian patterns.

3.2.1.1 Introduction

In this era social media encourage users to be active content producers instead of simple consumers, and users have the ability to share almost everything from anybody. Having the knowledge and a method to identify main producer of a content is an important asset and a difficult challenge. Previous studies have shown huge sharing activities of social media users [101], but a large portion of the content is simply copied/pasted from other accounts or sources without referring to their original publishers.

Undoubtedly, plagiarism detection in Online Social Networks (OSNs) is important, especially when the content belong to popular users (e.g. celebrities, politicians etc.) or major news agencies. This study attempts to identify the content originator of textual content in social media and to detect information propagation patterns among users based on their linguistic features and temporal behaviors.

In particular, content originator detection is critical in the context of fraudulent news and social media hoaxes. Some fake news increases readers tension while providing dangerous irresponsible information. For instance, false news stories circulated on social media played a major role during 2016 USA Presidential election campaigns [102]. In these scenarios, the foremost factor is to identify true identify of a user by manipulating the user's antecedent behavior. Moreover, if a shared social media post is anonymized, then user's online behaviors can be used to break the anonymity of these posts. Originality detection in OSNs based on unsupervised procedures are thus extremely important.

Authorship attribution is one key area that we can adapt to detect content originators. A large body of literature in author attribution has been proposed by utilizing users' writing styles, in which, the outcome revealed personal details such as gender [103], occupation [104], and age [105]. Towards that end, we implemented a framework manipulating user's writing patterns using the SCAP method [106] and users' online circadian behaviors. We then evaluate our framework using different test cases, with the goal of analyzing following research questions:

1. how efficient is the SCAP algorithm when applied to OSNs data (since the length of the text is limited) and what parameters do we need to consider in order to increase the accuracy of the system?

2. whether the circadian typology behavior of users in OSNs useful for detecting content originators?

3. is it possible to propose a framework for detecting content originators in social media by amalgamating users' writing style and online circadian typology?

3.2.1.2 Liturature Review

Many recent studies on authorship attribution of short and noisy text in social media have used machine learning techniques, NLP and similarity based approaches such as topic identification, genre identification etc. However, similarity based approaches outperform other methods when considering a large number of authors, a limited text size, and a large training set [107]. Referring to Table 3.1, many social network authorship attribution studies have used different similarity based mechanisms such as word and character n-grams, Source Code Author Profile (SCAP), Latent Dirichlet Allocation (LDA) and Author Topic (AT).

A number of Internet-scale authorship identification studies have focused on using a user's stylometric features in various disciplines such as, cyber-criminal detection [108], identifying the linkability of tweets [109], and cross domain authorship attribution in social media [110]. Additionally, some other works are based on analyzing a user's writing style

Table 3.1 – Previous works on authorship attribution in Social Networks

Technique	Classifier	Number of users [Ref]
Character level n-grams	SCAP	50 [112], 133 [113]
Character level n-grams	Naive Bayes	50 [114], 50 [115]
Character/Word level n-grams	SVM	50 to 1000 [7], 10 [116]
LDA	Topic model	[117]
LDA and AT	Topic model	274 [118], 1000 [119]

to detect fraud in digital forensics [111]. To date no studies have applied the circadian typology of users' online behavior.

3.2.1.3 Methodology and Proposed Solution

Compared to other web forums, OSNs comprised of more information on user-generated content (UGC), combined with a number of attributes such as timestamp, geo-location, content type (text, image or video), and user's personal details. This work is designed to identify the originator of a textual content in OSNs; the same idea can be applied to other content types as well. Since OSNs texts tend to have limited length, traditional author identification mechanisms are not accurate to determine originality [120].

There exists a number of author identification mechanisms for social media based on users' writing patterns. A limited number of studies have considered how temporal changes of user's writing style affects author attribution. Hosein et al. [113] identified that authors do change their writing styles at different time periods, and different authors change differently. Hence, they proposed a time-aware feature sampling approach by dividing an author's timeline into fixed-size periods to periodically analyze writing style. The proposed originality detection mechanism in our experiment aims to adapt users' temporal behaviors combined with their linguistic features in order to detect content originators of the content.

We propose an approach using four distinct phases for originality detection as depicted in Figure 3.1: (1) Data-extractor, (2) Pre-processor, (3) Feature-extractor and (4) Author-analyser.

1. **Data-extractor** : In this experiment, our main focus is on Twitter data and therefore, we use our Twitter crawler to collect tweets for a considerable period of time. The main attributes of the tweets that are useful for this experiment are the tweet text and the timestamp. All timestamps returned by Twitter streaming API are in ISO 8601 international standard with UTC timestamp offset in the format of YYYY-MM-DDTHH:mm:ssZ and therefore, the time is always guaranteed to be consistent.

2. **Pre-processor**: In this phase, our dataset is classified into different groups including texts, author details (i.e, username) and timestamp of the posts. We conduct our

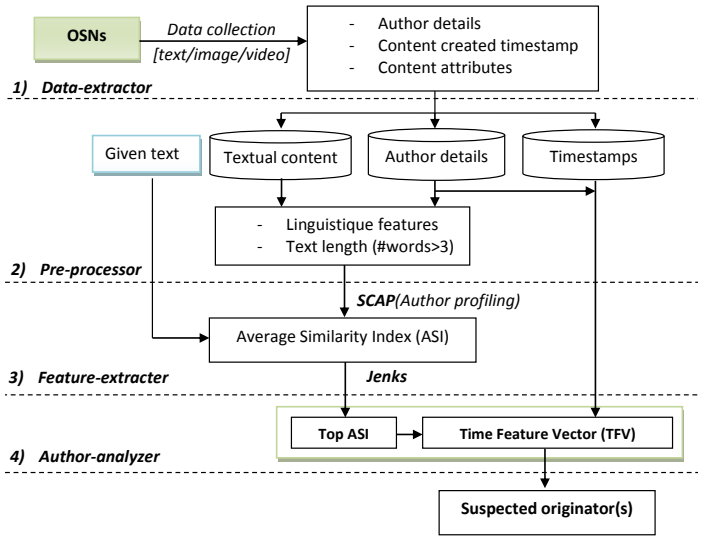


Figure 3.1 – ConOrigina: Content Originality Detection Framework.

experiments with both raw and pre-processed dataset in which we removed #tags, @username, URLs and filtered sentences with more than three words and we used only tweets in English.

3. **Feature-extractor**: The feature-extractor receives its input as a dataset which is classified in the pre-processor phase, and its output is a set of n-gram profiles for each user that are manipulated based on users' writing styles. Our study aims to explore the possibility of enhancing the outcome of an existing authorship identification mechanism, SCAP (Source Code Author Profiling) [106], one of the character-level n-gram approaches, was designed to identify the author of a computer program by profiling an author based on his commonly-used n-grams. In the SCAP approach, n-gram frequencies are considered as an authors' writing profile. These author profiles are used to examine similar writing styles of different users based on the intersection of their n-grams using Jaccard index. The higher the overlap measurements, the more similar those user profiles are.

   Despite the fact that the SCAP method is used to measure the overlap similarity of author profiles, applying directly to analyse a large corpus of short text is not very efficient [110]. Therefore, we examine the outcome of the SCAP approach by manipulating n-gram author-sub-profiles generated for a specific period of time. At first, the SCAP is executed for each user's tweets by reading a chunk (N lines) to generate author-sub-profiles (one sub-profile per chunk) and then produced n-grams (n in range(4,6)) for the provided text to produce a similarity score. Finally, we

calculated the author profile similarity in the feature extractor phase and returns Average Similarity Index (ASI) per user considering the similarity scores obtained for all author-sub-profiles with respect to a given text.

Algorithm 1 Originality Detection Algorithm

```
1: procedure ORIGINALITY(AUT, AKT, N, X)                          ▷
2: N: #text used in 1 SCAP execution
3: X: Set of top similarity index authors
4: AUT: author unknown text
5: AKT: author known texts
6:     if #words in AUT & AKT > 3 then
7:         while i = #authors do
8:             for k=0,j select N text of AKT_i do
9:                 out_AKT_k = SCAP(AUT, AKT_i)
10:                result.append(out_AKT_k)
11:            end for
12:            Similarity-Index.append(avg(result))
13:        end while
14:    end if
15:    X = Jenks(Similarity-Index)
16:    for each user i in X do
17:        if TT in TFV(AKT) then
18:            originators.append(i)
19:        end if
20:    end for
21:    return sort(originators) based on timestamp    ▷ 1st user in the list is the originator
22: end procedure
```

4. **Author-analyser**: The author-analyzer phase predominantly identify most adequate author(s) for a given text by utilizing the results from the feature-extractor phase.

   The Jenks natural breaks unsupervised classification and optimization algorithm is used to classify X number of users into the best cluster. The Jenks algorithm dynamically chooses each cluster according to the available dataset. We therefore do not set any specific threshold to cluster X users who have the maximum similarity scores identified in the Feature-extractor phase with respect to the given text. Furthermore, in this study, the value of the Goodness of Variance Fit (GVF) used in the Jenks algorithm is set to 0.8. Jenks algorithm returns one or more users that have the same writing styles with reference to the provided text. This experiment aims to assess the best matching author(s) for a given text based on the results obtained from the Jenks algorithm, as elaborated in algorithm 1.

   In the literature, very few studies have considered the temporal changes of users' writing patterns, and none of them have considered user behaviors in social networks

for author identification. Therefore, we can express our proposed methodology as a novel approach that characterizes users based upon their OSNs behaviors. We named this method as the *Time Feature Vector (TFV)*, as it utilizes the timestamps of all posts to identify the text distribution in different time periods. The *circadian typology* [121], the physiological behavioral measures, of a user are notably important when implementing TFV, particularly with regards to a user's social media routines. Generally, the circadian typology classifies individuals into three different types: morning, evening and neither. Within the social interaction framework, a morning type individual prefers to post in the morning hours while an evening type person posts in the evening. The foregoing discussion implies how we adapted the circadian behavior of an individual in online social network to identify content originators.

The TFV is applied on users who are classified as the top X users from the Jenks algorithm. In order to generate the TFV, first we inspect the frequency of number of posts shared in different time periods. We consider four time periods to categorize users according to their circadian behaviors (0-6hrs, 6-12hrs, 12-18hrs, and 18-24hrs). Then, for each time period, the relative frequencies of number of posts are calculated and represented the TFV as a vector of four elements. The social circadian behavior of a user is the time duration that belongs to the largest element in the TFV vector. The result of the TFV method is shown in algorithm 2. Finally, the timestamp of a given text is used to map with one of the circadian typology periods to identify the potential author(s) as presented in algorithm 1.

Algorithm 2 Time Feature Vector (TFV)

function TFV(AKT) ▷ n: #posts in AKT
  for `i=0, j=6, i+=6, j+=6` **do**
    if $i < T_{up} < j$ **then**
      $T_{up} \in F_{i,j}$
    ▷ T_{up}:timestamp of post-p of user-u and $F_{i,j}$:set of posts belongs to time period $i - j$hrs
    end if
  end for
  $TFV =< \sum F_{0,6}/n, \sum F_{6,12}/n, \sum F_{12,18}/n, \sum F_{18,24}/n >$
  return TFV[time duration of max index]
end function

3.2.1.4 Evaluation of the ConOrigina Framework

To evaluate our proposed method with the focus on exploring most effective parameters for SCAP method when it is applied on very short texts (tweets), and the usability of user's online circadian typology, the first step is to have a test dataset that includes similar posts in different user accounts. One of the communities that share many identical posts

Table 3.2 – Dataset Description

News Media	List of different categories of Twitter accounts of the News Media	Avg #followers	Total #Tweets
Reuters	Reuters, ReutersBiz, ReutersChina, ReutersIndia, ReutersLive, ReutersOpinion, ReutersPakistan, ReutersPolitics, ReutersTV, ReutersUK, ReutersWorld,LukeReuters	144.5K	38,756
BBC	BBCBreaking, BBCBusiness, BBCNews, BBCNewsAsia, BBCNorthAmerica, BBCSport, BBCWalesSport, BBCWorld	8.4M	25,747
CNN	CNN, cnnbrk, CNNent, cnni, CNNMoney, CNNPolitics, cnntech	1.9M	22,571
NYT	nytimes, nytimesworld, NYTNational, nytopinion, nytpolitics, NYTSports	1.1M	19,366
WSJ	WSJ, WSJOpinion, WSJPolitics, WSJSports, WSJTech, WSJusnews	391.3K	19,382
ABC	ABC, ABCPolitics	2.6M	6,463
SC	SportsCenter	31.6M	3,225

in social media is news media. In this respect, we considered seven popular news media Twitter accounts (legitimate publisher of the respective news media) and collected about 145K total number of tweets as shown in Table 3.2.

Evaluating the SCAP approach: To give a brief example of the efficiency of the SCAP approach, initially we considered a 100 sample tweets from *cnnbrk* (cnnbrk shows the highest number of active followers (47.3M) in our dataset). We observed that these sample tweets are originated from *cnnbrk*. Firstly, the SCAP method was executed dynamically between each test tweet (timestamp of the tweet:TT) and tweets published by all news media within a month (before and after TT). For each set of N tweets (N varied from 10 to 100), the SCAP method generates n-gram (n in 4,5, and 6) profiles (author-sub-profiles) that is used to measure the overlap n-gram similarity index with respect to n-grams profile of the test tweet. Figure 3.2 depicts, the precision of identifying a tweet posted by *cnnbrk* as a potential author of the test tweet for different N and n values. The results show that the precision of the SCAP is higher if 6-grams are used to build author-profiles and also, if user's author-sub-profiles are generated considering 10 or 20 tweets at a time. In view of that, the accuracy of identifying *cnnbrk* as a potential author among the list of users is 100%. We achieved this result after using pre-processed texts and for raw dataset, we achieved only 86% accuracy. This result indicates that the SCAP method is very efficient for detecting similarity of the short texts.

Evaluating the TFV approach (n=6 & N=10) For our sample dataset, on average, the SCAP+Jenks algorithm returns eleven identified users for the above experiment. In order to explore a best author, we evaluated the same dataset using our TFV approach. As described in the previous section, the TFV method uses circadian behavior of user's online activities. The circadian behaviors of many news media (49.01%) considered in this work belong to 12h00-18h00 duration, including BBCWorld, BBCBreaking, Reuters, ReutersWorld and 33.33% of them publish their content between 18h00-24h00, 11.76% of

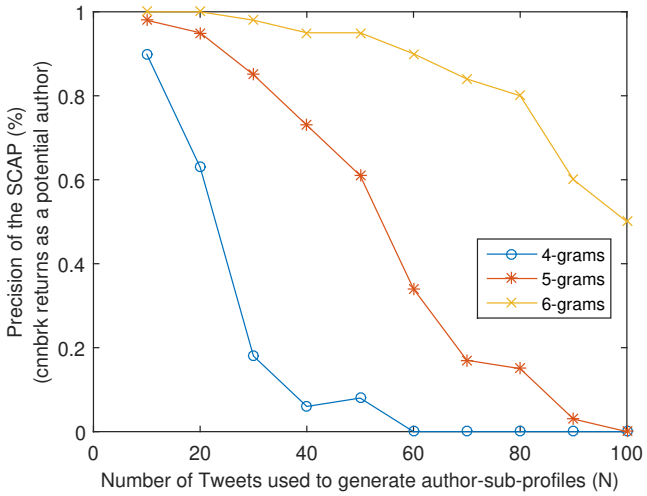


Figure 3.2 – Precision of the SCAP method for short texts.

Table 3.3 – Test Twitter dataset

Twitter account - #Tweets	**Precision%**	**Recall%**	**F1 Score**
BBC ***Injected Tweets: CNN - 10, Reuters - 10***	97.50	100.00	98.73
ABC ***Injected Tweets: CNN - 10, WSJ - 10***	97.50	95.12	96.30
Nytimes ***Injected Tweets: Reuters - 10, WSJ - 10***	95.00	95.00	95.00

Precision - TP/(TP+FP): the proportion of retrieved Twitter accounts that are relevant where retrieved originator and the considered Twitter account are identical. Recall - TP/(TP+FN): the proportion of relevant Twitter accounts that are retrieved. F1 score: harmonic average of precision and recall.

the Twitter accounts published during 06h00-12h00. After applying TFV method for the example considered in Figure 3.2, 6 users are classified out of 11 users. Hence, by using TFV classification on the SCAP result, we can filter the best potential authors (6 in this example) having similar writing patterns for a given text.

Another evaluation was performed using classic information retrieval metrics: Precision, Recall, and F1 scores, as shown in Table 3.3, based on 3 different test cases that are manipulated using 60 tweets. For each test case, we manually identified 40 tweets from BBCSport, ABC, and Nytimes separately. In the test dataset, 2/3 of the tweets are from these 3 main Twitter accounts and 1/3 are injected from other Twitter accounts, and hence, did not originate from the considered accounts. We manually checked the whole dataset to verify that the considered 60 tweets originated from the respective Twitter account. We performed the methodology described in this experiment and present the results in Table 3.3. The results indicate high precision and high recall parameters and the F1 score is more

than 95% for 3 test scenarios. In particular, only a small portion were classified as False Negatives due to the TFV method in which the timestamp of the considered tweet was not within the circadian typology period of the originated account. Therefore, classification shows the injected Twitter account as the originator in those cases. In addition, we also experimented with 3224 tweets shared by SportsCenter and the proposed approach could detect about 89% of the tweets shared in its Twitter account as its originated tweets. This is mainly due to SportsCenter publishes news related to sports and other news media users consider in our work are general news publishers.

3.2.2 Experiment 2: News Originators and Consumers of News Media in Social Media

The second experiment also uses text similarity measures to understand news dissemination patterns in social media. Most of the analysis explain in this section are based on the ConOrigina framework proposed in the Section 3.2. In this experiment, we collected news items from 48 different news media from both Twitter and Facebook.

3.2.2.1 Introduction

The rise of online social networking platforms has transformed the news industry in many unforeseen ways. As per the research done by Pew research center in 2018 [122], about two third of American adults get news on social media sites and most commonly they used Facebook followed by YouTube and Twitter as social media sites. News media in social media allow their audience not only to interact(shares, likes and comments) with the content but also via news creations [123]. Excessive prominence of the news media in online social media sites vastly depend on its overall popularity and audience engagements. In this perspective, one way for a news media in this era to get ahead in the news media community is to attract more followers.

An interesting finding of Alice et al. supported the claim that distribution of newspaper content through social media sites are becoming a common practice and comparatively, Twitter is more effective than Facebook in terms of audience reach [124]. In addition, Another study has shown that social media sites such as Twitter use textual content as the most common format of information that news media use as their source [125]. Therefore, it is worth to analyze the textual posts published by news media in Facebook and Twitter to classify based on different features including news production behaviors, news dissemination behaviors, and news reader reactions. These analyses help to determine news leaders and providers among news media. To identify news leaders and providers in the news media community, one influential factor we can observe is their content features. For instance, we can consider content features as the originality of a news item (who created the post), that

is evaluated based on the linguistic characteristics such as the writing style of a news media. In addition we can consider context features (time of creation) and news item popularity among readers to explore news leaders and providers in the social media.

In Experiment 3.2, we presented a conceptualized framework *ConOrigina* to identify content originators of textual contents in social network sites [126]. ConOrigina detects a content creator by leveraging observable information such as their linguistic features and temporal behaviors (circadian typology) to identify user-content relationships. We explored that the consideration of temporal changes in user behaviors in social media has significantly improved the detection of content ownership. User-content relationship in social media helps to identify users with similar writing styles and identical temporal patterns. We can apply ConOrigina to detect news item originators in news media community in Facebook and Twitter. In this scenarios, user-content relationship can be interpreted as; user - news media and content - news items. In this experiment we use ConOrigina to inspect news media synergies in social media in-terms of different metrics;
i) News self-originators: news media who originate content by themselves or publish fresh content,
ii) News providers: news media who distribute their content to other news media, and
iii) News consumers: news media who mostly share replicated content that have been already published by other news media.
Apart from that, this experiment explore news originators in Facebook and Twitter, and cross-posting activities in between news media in Twitter and Facebook. We will also identify news media popularity parameters and circadian behaviors of popular news media in order to explore their publishing patterns. Finally, we propose a reader reaction predictive model for news media to increase news popularity based on the considered features. Moreover, unlike in the literature, we considered news media that are popular worldwide rather than choosing from a specific geographic location. To that end, we used data mining, text mining and statistical methods to perform descriptive and predictive analyses on the dataset crawled from 48 news media presence in Facebook and Twitter. The descriptive analyses mainly use the result from ConOrigina framework to explore news origination patterns, news dissemination patters and news popularity parameters coalescing both content features and contextual features. Then, predictive analyses examine how contextual features and historical behaviors (number of published posts, number of followers) can combine to predict reader engagement in future.

3.2.2.2 Literature Review

Traditional news providers are loosing their traditional income streams due to arrival of social networking services allowing any user in the network to share and propagate news

articles freely and easily. Therefore, in order to survive those news media, they need to understand the dynamics of news production, dissemination and consumption in social media. Social media sites such as Facebook, Twitter and YouTube has become popular venues for news media sites [122]. As a result, insight into the news shared in social media helps to understand the impact of news on readers as well as news media community. Many previous works focus on the dynamics of news on social media, mainly in Twitter, that impact the attention and engagement by news readers. Hence, in this section, we will provide the main findings of the literature and the features adapted in their studies to analyze the news media in social media focusing on different aspects.

During 1990 major news providers moved online, including The New York Times and The Washington Post [127]. Then, during 2000 news media have fully moved in to the online portals providing digital access [128]. The emergence of social media sites such as Facebook (2004), YouTube (2005), and Twitter (2006) was another major turning point of news industry that provide ease of news delivery and provide an opportunity for the audience to interact with the news items by means of sharing, writing, giving feedback (commenting) etc. In America and Canada, majority of the adults receive news on social networking sites [122] [129]. Of all the social media sites, Facebook has emerged as the top news receiving site followed by YouTube and Twitter [122]. Some studies [124], [125] claimed that even though Facebook has more users than Twitter, the latter is more widely used as a source of news. In particular, Twitter has been identified as a source of breaking news [130] [131] and these breaking news get attraction from the readers before other social media sites like Facebook and Google Plus [132]. Hence, in the following sections we survey how news papers and news media use Facebook and Twitter to disseminate news and how can those news media can increase the news readers' attention.

The news posts are different to the other types of the posts in social media in which their target is on personal details, events and jokes. Hence, previous research studies focus on which attributes need to be considered to get the attention from the news readers using real datasets from social media. News posts typically contain links to the news article or small reports on a event. These news posts has three different features including content features (what is written), user features (details on the who is posted) and context features (time, location). Literature addressed on various analysis such as, analysis of the life cycle of online news stories [133], predicting the popularity of the news [134] [135], topics discussed in the news posts [136].

Since different readers have different opinions on news posts, previous studies have tried to capture readers' engagements on news items and news categories. Orellana-Rodriguez et al. [136] [137] have shown that sports news readers responded differently at different times of the day and week than business news item readers. Another study [138] has

analysed how news in social media was different to main stream media and explored that there exists gaps between what the news media publish online and what the community shares in social media. They showed that ordinary events receive more coverage in Twitter than in mainstream media. Another study has suggested that user interest on topics is a good predictor for a focused attention and positive affect [139]. Therefore, they proposed to enhance diffusion of user interested topics for boosting user engagements. Many news providing user accounts or journalists try to have a rich follower count aiming to gain high attention and engagement [140]. However, in another study they suggested that having a large number of followers is not sufficient and followers must be active spreaders of these news items [141]. In addition, some other research studies have considered context of the news item posted in social media such as temporal aspects and location aspects. For instance, some studies found that the audience engagement is different for news categories at different days in different times [142]. There are some works that considered all the features related to the content, user and context. For instance, [136] [143] revealed that different combinations of features influence reader engagement differently with various news categories. Moreover, they have used regression analyses to predict user engagement based on content, user, and context features.

We can find different perspectives in the previous work to analyze news content in social media using the features from user, content and context, and few of them have adapted all features together. There is no any deviance on the fact that analyzing news media in terms of the news item originators perspective. Our study combine features from content and context to get very useful insights from news media presence in worldwide. As mentioned earlier, since news media tend to use social platforms to share their content, it is worthwhile to analyze which news media is propagating the content first in the social media and becoming popular among users with regard to readers' interactions.

3.2.2.3 Dataset description

This section begins by describing the dataset used in this experiment aiming to identify behaviors of news media in terms of their published content.

Data Collection: The analysis of this experiment is based on a dataset collected from top 48 most popular news media (English edition) ranked by Alexa[1] retrieved on 2nd May 2017. Firstly, a set of active and authorized user-IDs and their account names (135 user-IDs) were manually identified from Facebook and Twitter. Subsequently, for each user-ID, Facebook and Twitter crawlers were executed to extract time-line posts and their respective timestamps within one month period starting from 8th May to 8th June 2017. We believe that the posts published during one-month is sufficient to analyse the propagation patterns

[1] http://www.alexa.com/topsites/category/Top/News

Table 3.4 – Dataset description for multiple user-ID scenarios and single user-ID scenarios in the news media dataset.

		#Followers(avg)	#Posts
Multi user-ID (135 users)	Twitter	6570K	238K
	Facebook	8312K	128K
Single user-ID (48 users)	Twitter	2836K	152K
	Facebook	3630K	80K

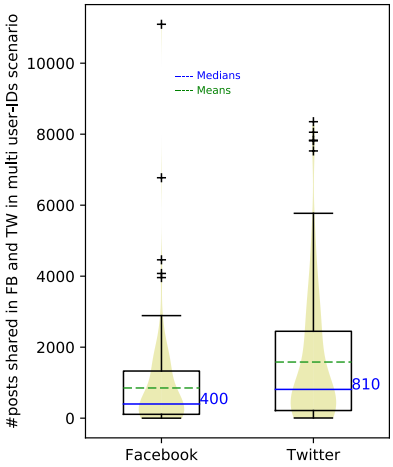


Figure 3.3 – Multi-user-ID scenarios

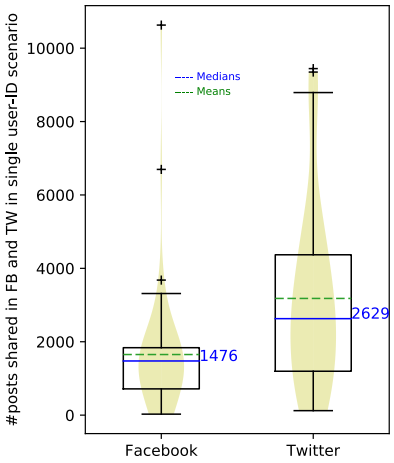


Figure 3.4 – Single-user-ID scenarios

Figure 3.5 – Distribution of the number of posts shared by 48 popular news media in Facebook and Twitter during 8th May 2017 to 8th June 2017.

of news content as news media tend to share news items frequently in the social media. Since Twitter RESTful API allows only to retrieve last 3200 posts, data collection is started on 8th May 2017 and the crawler is executed every 5 days to retrieve a full dataset of one month from 8th May to 8th June 2017. However, during that period, Facebook graph API was allowed to extract any public post and hence the Facebook crawler is executed only once. During the crawling period, in order to respect the ethical aspects we collected only the public data from the news media's social media accounts and not collected neither sensitive data nor personal data.

Table 3.4 shows a brief description about the dataset acquired from Facebook and Twitter. A handful of news media has multiple user-IDs to represent different categories (technology, politics, breaking news, etc.), and others use only one legal user-ID. Therefore, we consider two different scenarios to explain the dataset; multi-user-ID scenarios (exhibits multiple user accounts per news media) and single-user-ID scenarios (exhibits a single user account per news media).

In Multi-user-ID scenarios, we look attentively at the full distribution of the dataset belongs to the news media considering total of 270 users selected from the Facebook (135 users) and Twitter (135 users). In Single-user-ID scenarios, dataset consists of 96 users selected from the Facebook (48 users) and Twitter (48 users) where for some news media a single user-ID is chosen based on the highest number of followers compared to other user-IDs identified from the same news media assigned for different categories. Following that, Table 3.4 includes a brief description of the dataset for both Multi-user-ID and single user-ID scenarios separately for Twitter and Facebook. It gives the information about average number of followers in each scenario and total number of posts shared by all the news media in both Twitter and Facebook. To better explain the distribution of number of posts shared on Twitter and Facebook for two scenarios, we generated violin plots shown in Figure 3.5. These plots illustrate the abstract representation of the probability distribution of the dataset based on the symmetrical kernel density estimation (KDE) showing the variability of data.

Multi-user-ID scenarios: Total number of posts crawled from Twitter and Facebook in the multi-user-ID scenarios are 238K and 128K, respectively. In spite of the fact that highest number of followers belong to Facebook user-IDs compared to the same set of user-IDs on Twitter, number of posts shared by them in Facebook is lower than number of tweets. Some preliminary work carried out in the recent years has also proven the same that Twitter has been widely used as a source of news than Facebook [124], [125]. The distribution of number of posts shared in the multi-user-ID scenarios, after being pre-processed, is depicted in Figure 3.3. Many posts were shared by the Twitter user-IDs, on average, which is as twice the rate of shared Facebook posts. According to the observations, no significant differences were found on the number of posts before and after pre-processing. However, on average, more tweets were removed after pre-processing than Facebook posts. One possible reason is that, many tweets shared by some news media had lengths less than 3 words.

Single-user-ID scenarios: As mentioned earlier, Single-user-ID scenarios consider only one user-ID per news media, the most popular Twitter and Facebook user of the news media with highest number of followers compared to other users who represent the same news group. In total, we extracted content of 96 user-IDs (48 each from Facebook and Twitter) with additional information such as number of posts shared and number of followers as detailed in Table 3.5. The violin plot shown in Figure 3.4 shows the distribution of pre-processed dataset from single-user-ID scenarios, indicating that less number of posts shared on Facebook than on Twitter. In the Facebook dataset, number of posts ranges from 0 to 3.7K (median: 1476) and that of Twitter is from 0 to 9K (median: 2629)and the medians of both distributions reside in the higher probability range of the KDE. Apart from that, Figure 3.4 indicates an outliers, a single Facebook user-ID, as having the largest

Table 3.5 – Dataset (single user-ID scenarios) of the list of the most popular news media worldwide, ranked by http://www.alexa.com on 28 April 2017.

News Media (#users)	**Account name**	**#posts**	**#followers 10 ^5**	**Account name**	**#posts**	**#followers 10 ^5**
Accuweather.com	AccuWeather	721	14.91	accuweather	122	8.48
Aljazeera.net	aljazeera	896	96.28	AJEnglish	2745	39.30
Alternet.org	AlterNetNews	469	9.59	AlterNet	293	1.34
Ap.org	APNews	321	3.29	ap	3220	37.35
Bbc.co.uk	bbcnews	1056	39.47	BBCNews	2689	8.70
Bloomberg.com	bloombergbusiness	1913	6.49	business	9349	8.73
Cbc.ca	cbcnews	1333	4.6	CBCNews	3838	6.09
Cbsnews.com	CBSNews	2589	20.81	CBSNews	1892	30.35
Chron.com	houstonchronicle	346	0.26	HoustonChron	1921	4.80
Cnbc.com	cnbc	1953	20.49	CNBC	3247	26.60
Cnn.com	cnn	1579	47.25	CNN	4697	40.30
Dw.com	deutschewellenews	633	4.02	deutschewelle	182	0.38
Economictimes.com	EconomicTimes	2412	20.2	EconomicTimes	5284	13.43
Euronews.com	euronews	1846	16.85	euronews	1752	2.58
Forbes.com	forbes	699	47.11	Forbes	2026	128.00
Foxnews.com	FoxNews	1421	33.04	FoxNews	4737	22.05
Hindustantimes.com	hindustantimes	6696	18.69	htTweets	8790	19.46
Hollywoodreporter.com	HollywoodReporter	1606	22.14	THR	5399	25.80
Huffingtonpost.com	HuffPost	1421	36.28	huffpost	891	26.87
Indianexpress.com	indianexpress	10629	68.38	IndianExpress	8724	24.70
Indiatimes.com	indiatimes	1896	69.64	indiatimes	1709	0.46
Ipsnews.net	ipsnews	74	2.2	ipsnews	245	0.18
Nationalgeographic.com	natgeo	324	124.22	NatGeo	427	65.34
Nbcnews.com	NBCNews	1588	17.43	NBCNews	2683	8.87
News.com.au	news.com.au	1530	8.57	newscomauHQ	2875	4.70
News.yahoo.com	Yahoo	1314	105.64	Yahoo	1068	13.05
Newswise.com	Newswise	34	0.02	newswise	1239	0.12
Nypost.com	NYPost	1250	39.64	nypost	2425	12.70
Nytimes.com	nytimes	1668	30.48	nytimes	3659	97.44
Reddit.com	reddit	27	11.07	reddit	155	4.84
Reuters.com	Reuters	1499	11.02	Reuters	3805	37.49
Smh.com.au	sydneymorningherald	1643	3.11	smh	1808	2.56
Theatlantic.com	TheAtlantic	1453	21.38	TheAtlantic	2473	15.40
Thedailybeast.com	thedailybeast	1191	22.08	thedailybeast	4486	10.80
Theguardian.com	theguardian	1617	74.49	guardian	8601	61.50
Thehill.com	TheHill	1635	11.83	thehill	9443	24.20
Thehindu.com	thehindu	1113	51.28	thehindu	2514	42.30
Time.com	time	1509	120.02	TIME	5136	140.00
Timesofindia.com	TimesofIndia	3314	59.16	timesofindia	6052	53.55
Upi.com	UnitedPress	172	0.75	UPI	873	0.41
Usatoday.com	usatoday	2589	82.85	USATODAY	4331	33.40
Usnews.com	usnewsandworldreport	992	10.91	usnews	2575	1.26
Washingtonpost.com	washingtonpost	1743	58.27	washingtonpost	3996	102.00
Weather.com	TheWeatherChannel	3680	77	weatherchannel	997	28.10
Weather.gov	NWS	84	2.52	NWS	353	6.99
Wsj.com	wsj	1835	9.82	WSJ	3724	25.65
Wunderground.com	wunderground	146	3.94	wunderground	732	3.86
Xinhuanet.com	XinhuaNewsAgency	2813	195.71	XHNews	2463	91.90

number of posts in the entire dataset.

Analysis of this work is based on the Single-user-ID scenarios. Table 3.5 shows the considered user-IDs for each news media, number of posts shared by them and its number of followers. In order to visualize the distribution of posts in the Single-user-ID scenarios, we constructed a bar graph shown in Figure 3.6. A closer look at the distribution of this data indicates that the highest number of posts (10.6K) were shared by the Facebook user-ID of *Indianexpress*, followed by *Hindustantimes(6.7K)*. In contrast, five news media in Twitter were posted more than 85K posts during the crawling period: *Thehill, Bloomberg, Hindustantimes, Theguardian, and Indianexpress*. These high numbers of shared news items on Twitter and Facebook indicate that *Indianexpress* is more active on both social networks compared to other news media. On the other hand, four news media were published less than 100 posts on Facebook (the least number of posts were shared by *Reddit*) while all news media in Twitter were shared more than 100 tweets.

In conclusion, based on our inspections, 77% of the news media were active on Twitter than on Facebook as manifested in the previous works [124] [125], and only two news media, *Indianexpress* and *Hindustantimes* were used both social networks very actively. A few others, *Reddit and Ipsnews*, were shared few news items on Twitter and Facebook, and remainder mostly used either Facebook or Twitter exclusively.

3.2.2.4 Descriptive analysis of news media posts

News media presence in Facebook and Twitter are likely to share syntactically and semantically coherent news items. As a result, it is very challenging to detect who produce the content first in social media platforms. It is not sufficient to use a simple similarity matrix and the timestamp of news items to identify news producer of each news item. For instance, if a news producer removes the content from social media there is no any evidence to prove the real content originator. As a consequence we can consider writing styles of the news media, as a write print, to distinguish users separately and also their publishing patterns or the circadian pattern to analyze news producers and consumers in the news media community in social media. Identifying their publishing patterns such as most active time and day for each news media provides more information about their behaviors.

This section describes news media behaviors in Facebook and Twitter with reference to following 3 measures:
News origination - those who share fresh news content and those share replicas,
News dissemination - news propagation patterns and interactions among news groups,
News popularity - popularity of news items among news readers and how news media can enhance their popularity using different new features consider in this work.

In order to identify news origination and dissemination patterns, we first need to dis-

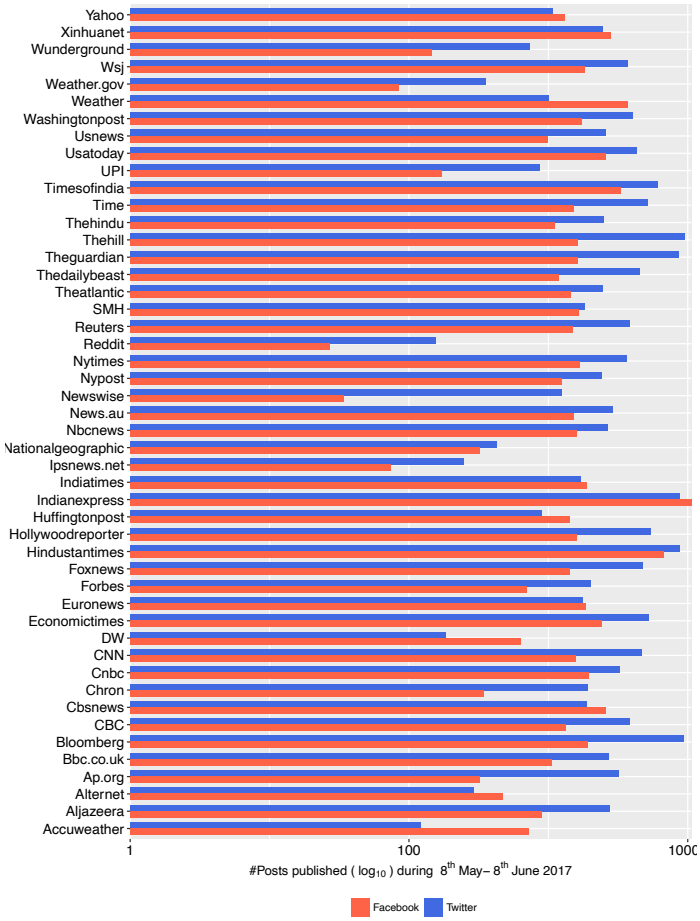


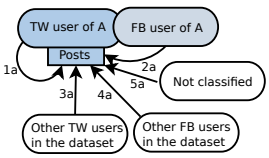
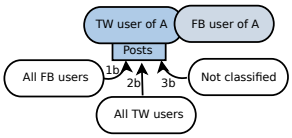
Figure 3.6 – This bar graph depicts the distribution of number of posts published by top 48 news media. Indianexpress(10.6K) and Hindustantimes(6.7K) were shown to be the news agencies that published highest number of posts in Facebook. Most widely published tweets were belong to Bloomberg(9.4K), Thehill(9.4K), Hindustantimes(8.8K), Indianexpress(8.7K), and Theguardian(8.6K).

cover news originator of all the news items. Detection of a content originator for a given text is an important asset to observe information propagation patterns among different news media that helps to explore news replicas shared by other news media. In that regard, the framework ConOrigina in Section 3.1 is capable of detecting content originator of news items [144]. ConOrigina can detect similar textual content considering the writing patters of the user.

News origination: The identification of news originators in social media is important to understand who are the content producers and consumers in the news media community. News originators generate fresh news items and publish first in the social media. Therefore, this section aims at analyzing the originality of content posted in Facebook, in Twitter and in between Facebook and Twitter (cross-posting), and then attempts to identify news origination patterns of news media in Twitter and Facebook.

At first, ConOrigina [144] framework is executed on the pre-processed dataset explained in Section 3.2.2.3, obtained from top 48 news media users. To better understand news media behaviors in social media, we amalgamated tweets and Facebook posts together, in total of 232K posts, to detect the first owner of each post using the ConOrigina. We use two different content transition approaches to elaborate news production behaviors of news media. Table 3.6 shows these transition diagrams and their descriptions.

Table 3.6 – State-Diagram-a (SDa) and State-Diagram-b (SDb) depict different transition patterns of the news media content with regard to their content originality. In this example, we assume that the posts were originated by the Twitter user of a news media A and similarly we can consider for the news media user in Facebook.

	Transition diagram	Link	Description: Portion of the -
SDa		1a 2a 3a 4a 5a	content in A that are originated by itself. replicas in A that are originated by the respective FB user. replicas in A that are originated by the remaining TW users. replicas in A that are originated by the remaining FB users. content in A that might not be able to classify into any other state.
SDb		1b 2b 3b	replicas in A that are originated by all FB user-IDs (including A). replicas in A that are originated by all TW user-IDs (including A). content in A that might not be able classify into any other state.

The **diagram SDa** exhibits five different states and transitions representing all possi-

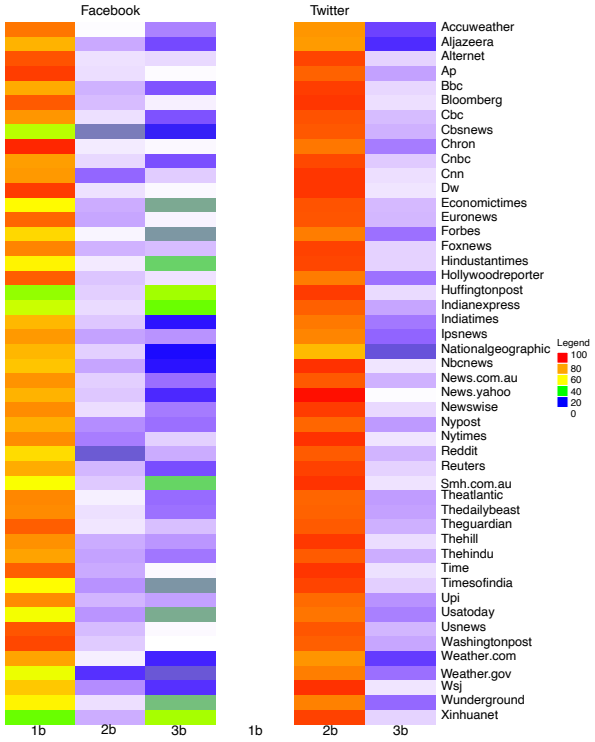
bilities that content of a particular news media might belongs to. These transition patterns are identified in terms of the originality of shared tweet or Facebook post. For example, in SDa news media A possess a set of posts shared by its own Twitter user-ID and we can elaborate five different content originality transitions as follows;
1a: portion of the self-originated content of Twitter user of A,
2a: portion of the replicas that were initially originated by the Facebook user of A,
3a: portion of the replicas that were originated by other news media in Twitter,
4a: portion of the replicas that were originated by other news media in Facebook, and
5a: set of posts that are not classified. The sum of the weights of the transition links should be equal to the total number of posts shared by Twitter user of A.

The **diagram SDb** shows content transitions belong to Twitter user of A in another perspective: total number of transitions from all the users in Facebook and in Twitter separately. Diagram SDb helps to quantitatively understand whether the portion of content shared by Twitter user of A was originated by the users presence in Twitter and/or in Facebook. Total number of post shared by the Twitter user of A can be identified using following transition links;
1b: portion of the content originated by all Twitter users,
2b: portion of the content originated by all Facebook users and
2c: set of posts that are not classified.

In order to pictorially represent above state transitions within 48 news media, we built two color-maps, separately for Facebook and Twitter transitions, as shown in Figure 3.7. Figure 3.7a detailed the information of SDa (the five possibilities on how a content was originated) and Figure 3.7b represents SDb state transitions. This approach was designed to help us better understand news creators in social media and their interactions by investigating how many posts are originated and published by each news media.

News originators in Facebook: In Figure 3.7a SDa state transitions of news media in Facebook are indicated under the 'Facebook' label and therefore label 2a represents the portion of content originated by its Facebook user itself. As shown in Figure 3.7a, almost all news media (about 96%) accounts in Facebook have shared a higher portion of content originated by themselves than replicas. An interesting observation is that, in our dataset, *Reddit* and *Cbsnews* have shared less self-originated content and more replicas, and it is more visible in Figure 3.7a. Another observation is that, 49% of the *Huffingtonpost* news items are non-classified, 36% of the posts are self-originated in Facebook, and 18% of replicas that were first published by its own Twitter user account. The main reason for classifying many Facebook posts belongs to *Huffingtonpost* as non-classified is that the lengths of many posts published by *Huffingtonpost* are only 4-6 words, which is smaller when compared with

(a) News originality transitions with reference to the transition diagram SDa in Table 3.6



(b) News originality transitions separately from all Twitter and Facebook user accounts with reference to the transition diagram SDb in Table 3.6

Figure 3.7 – These two figures clearly depict different transition patterns of the news media content with regard to their content originality.

the posts of other users.

Overall, a majority of the news media have shared their self-originated content in Facebook, while a few others have published replicas that are very similar to the posts originated by the remaining news media.

News originators in Twitter: In Figure 3.7a, column labeled as 'Twitter' indicates SDa state transitions representing news media users in Twitter. An interesting observation is that, 60% of the news media in Twitter have exhibited content transitions from other Twitter users (link 3a in SDa), which means that many Tweets are replicas that were initially posted by other news media in Twitter. In other words, only 40% of the news media in Twitter published a higher proportion of self original posts (link 1a in SDa) than replicas (link 3a and 4a in SDa).

Based on the results of the ConOrigina, a set of news media in Twitter was originated more than 90% of their own content: *Cnn, Dw, News.yahoo, and Newswise*. On the other hand, another set of news media was originated less than 10% of their published posts: *Usatoday, Indianexpress, and Ap*, indicating that they were mainly posted replicas. Moreover, we examined that none of the news media has posted replicas that were initially published by any news media in Facebook where this is illustrated in Figure 3.7a with two sub-columns labeled as 2a (number of posts originated by the same news media in Facebook) and 4a (otal number of posts originated by other news media in Facebook respectively).

To summarize, news media in Twitter have not shared replicas that were originated by any news media in Facebook. Hence, all the news posted in Twitter were originated by news media users in Twitter (links 1a and 3a in SDa) and many news media in Twitter have shown transitions from other news media in Twitter indicating that Twitter news media user-IDs have posted replicas of other Twitter news media user-IDs .

Cross-posting activities: As shown in Figure 3.7b, news media users in Twitter do not exhibit any transition from Facebook users (represented as a white column). However, news media Facebook user accounts disclose transitions from other news media users in both Facebook and Twitter. As a result, an interesting aspect we identified from this observation is the cross-posting behaviors. In this study, term cross-posting is used to analyze news replicas shared across Facebook and Twitter. Figure 3.7b illustrates that none of the news media in Twitter has shown cross-posting activities, while in Facebook we can observe a significant cross-posting behavior. Therefore, we quantify the cross-posting behavior of news media by focusing only on Facebook posts and the transitions. To accomplish above mentioned quantification, we compare how many cross-posts were generated
i) in between Facebook and Twitter user of a news media and

Figure 3.8 – Cross-posting activities of the news media are analyzed based on their Facebook posts. This graph illustrates the portion of the cross-posts shared by Facebook user-IDs that are originated by Twitter user-IDs

ii) among one Facebook user and all other Twitter user-IDs of news media.

We identified that average number of cross-posts in between a Facebook user and its respective Twitter user is as 4, and the similar figure for a Facebook user and all other Twitter users is 3. Consequently, one news media in Facebook was able to exhibit at least 3 cross-posts within the remaining Twitter users and 4 cross-posts in between its respective Twitter user.

It is worth to understand whether a Twitter user of a news media is the real cross-post originator as this leads to conclude that the cross-news posted in Facebook were originated by its Twitter user-ID. To determine this characteristic, we generated a graph shown in Figure 3.8 using normalized values of the number of cross-posts between Facebook and Twitter users. In Figure 3.8, the diagonal represents news media those who exhibits cross-posts between their own Facebook and Twitter user account. A small number of news media

namely, *DW, Wunderground* and *Alternet*, do not exhibit any cross-posts between their own Facebook and Twitter users. *Cnn* disclosed the largest portion of the cross-posting behavior (1K posts) in which, approximately 13% of its Facebook posts were exactly similar to its tweets. Besides that, about 70% of the cross-posts of *Cnn* in Twitter was shared with the Facebook user of the *Cbsnews*, which is equal to the 13.8% of the total posts shared by *Cbsnews* in Facebook. Another interesting observation from Figure 3.8 is that, Twitter user-ID of *Time* had shown cross-posts among 40 other news media in Facebook. The second-largest number of cross-posts in our dataset belongs to *Time*, in which, majority of the cross-posts were shared within its own Facebook user (about 15.7%). Other than this, many other news media have exhibited a very limited number of cross-posts in between other Twitter user-ID of news media.

To conclude, based on the observations, main cross-posts contributor for a news media in Facebook is its respective Twitter user-ID. This indicates that, many news media first publish their news item in Twitter and then post the similar content in their Facebook account.

3.2.2.5 News dissemination

The diffusion of news in social media relies on the propagation behaviors by different actors (news media). News dissemination process can be cyclical, dynamic and self-referential. These propagation relationships in social media form a complex network with regards to the content diffusion patterns. Therefore, understanding news dissemination patterns of the news media in Facebook and Twitter can be performed through analyzing the dissemination network. Our definition of news dissemination refers to propagating one news item from a news originator (news media who originated the news item) to the consumer (news media who shared an exactly similar copy of the news items that were initially posted by others). In this section, we explain some of the inter news media news dissemination and interaction patterns.

The news dissemination graph among news media users are shown in Figure 3.9. In which, the sectors correspond to the news media user account in Facebook and Twitter. The width of a sector is proportional to the number of interactions that a particular news media performs with others. Grid colors which represent sectors, and link colors that are the same as the grid colors represent different news media interactions in Facebook and Twitter. The position of the starting end of the link is always shorter than the other end to show that the link is moving from one news media and being received/consumed by another news media. Additionally, the large arrows in the graph indicate the direction of the information flow. The names of the news media are sorted in alphabetical order for ease of comprehension. News media in Twitter are labeled in blue and Facebook user-IDs

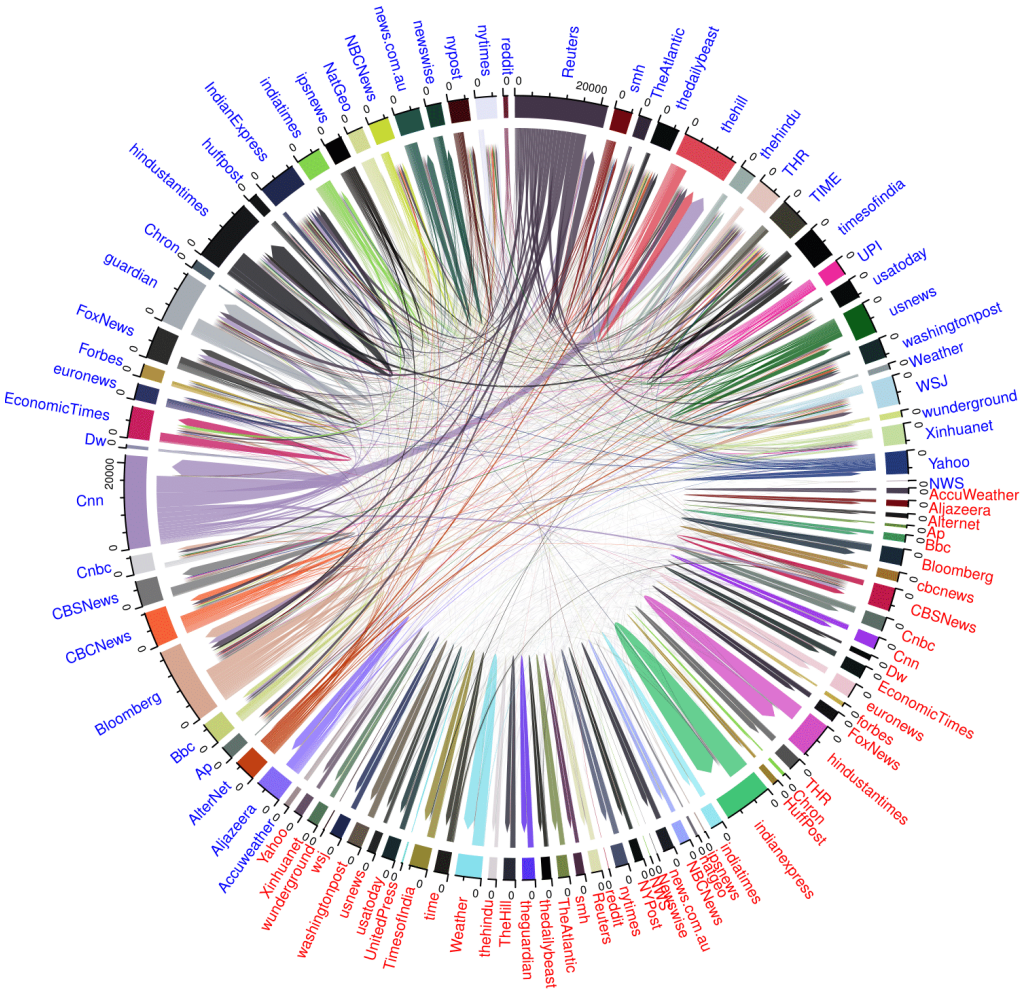


Figure 3.9 – News propagation patterns and news media interactions among 48 different news media

are labeled in red.

To evaluate interactions between news media in Facebook and Twitter, we consider three different measures in this analysis. For example, we can elaborate three measures for a news media A:

- *SelfLink* $(A \rightarrow A)$ - portion of the content originated by A;
- *DispersedLinks* $(A \rightarrow B)$ - portion of the content originated in A that are dispersed to B;
- *AcquiredLinks* $(B \rightarrow A)$ - portion of the content acquired by A that are originated in B (reverse of DispersedLinks).

We can build a model to convey the number of interactions ('In') performed by one news media in Facebook or Twitter as follows where, W stands for the weight. The sector sizes in Figure 3.9 indicate the value of 'In' for each news media separately for Facebook and Twitter.

$$\textit{'In'} = \sum(W_{SelfLink} + W_{DispersedLinks} + W_{AcquiredLinks})$$

As depicted in Figure 3.9 majority of the news media in Twitter have shown the largest 'In' compared to their Facebook users. We can deduce from Figure 3.9 that news media are more active in Twitter than in Facebook in terms of total interactions performed with others, and that a majority of them have shared self-originated content on Facebook having a limited number of interactions with others in Facebook or in Twitter. This is reasonable, since the largest number of posts were shared by the news media in Twitter.

Among all news media in our dataset, highest 'In' values, more than 18K, were performed by *Reuters* and *Cnn* in Twitter. The AcquiredLinks of these two news media compared with those of others are negligible. For instance, weight of AcquiredLink of *Reuters* (Cnn → Reuters) is 3.3% from its 'In' and, that of *Cnn* (Reuters → Cnn) is only 0.21%. Therefore, Cnn and Reuters were more active in Twitter as they have shown highest 'In' values. However in Facebook, largest 'In' value appeared from *IndianExpress*, about 6K. As a result, we can conclude that *IndianExpress* is the most active news media in Facebook.

Following sections will explain strong news providers and strong news consumers with reference to Figure 3.9 considering the portion of DispersedLinks and AcquiredLinks. In this section, the analysis on SelfLink is not much important as it provides the information about number of posts originated by a news media that we have analysed in Section 3.2.2.4

Strong news providers: A strong news provider exhibits a large portion of number of DispersedLinks than the portion of SelfLink and AcquiredLinks. They are the content leaders in the news media community in social media.

Based on our analysis, only 10 news media in Twitter have exhibited higher weights for DispersedLinks having greater than 70% of its total interactions. These are *Accuweather, Alternet, Cnn, Dw, Ipsnews, Nationalgeographic, Reddit, Reuters, UPI* and *Weather.gov*. This set of news media can be considered as the content providers in Twitter. In particular, *Ipsnews* and *Alternet* were exhibited greater than 95% of its total interactions as DispersedLinks idicating almost all of its links as outward links in Figure 3.9. A notable relationship in Figure 3.9 is that the largest DispersedLink is shown in between Twitter user-IDs of Cnn and TheHill (cnn → TheHill). This indicates that 42% of the 'In' values or total interactions of TheHill were performed with Cnn, which further conveys that the majority of the posts published by TheHill were replicas that were initially shared by Cnn.

In contrast, there is no any important relationship among news media in Facebook in terms of their DispersedLinks.

To sum up, the top 10 news providers of most popular 48 news media are *Accuweather, Alternet, Cnn, Dw, Ipsnews, Nationalgeographic,Reddit, Reuters, UPI* and *Weather.gov*.

Strong news consumers: The term strong news consumer is used to classify news media those who shared many replicas (shows higher weights for its AcquiredLinks than DispersedLinks and SelfLink).

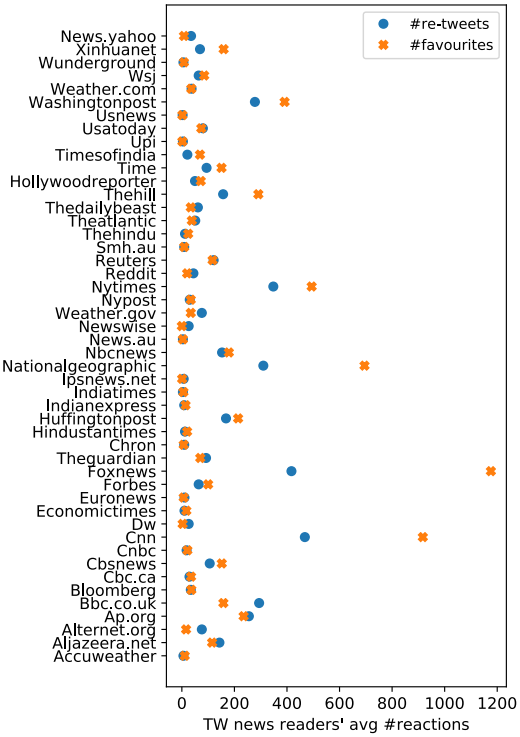
As shown in Figure 3.9, small number of news media have presented a large weights for AcquiredLinks with respect to their DispersedLinks. Based on our observations, top 10 content consumers in Twitter those who have exhibited greater than 70% of its total interactions as its AcquiredLinks are: *Ap, Chron, Hollywoodreporter, Indianexpress, Timesofindia, Nytimes, Theatlantic, Thedailybeast, Usatoday*, and *Washingtonpost*. Among them *Ap, Usatoday* and *Indianexpress* were exhibitted more than 82% of total interactions as its AcquiredLinks. Moreover, *Ap* was revealed the least number of weights for its DispersedLinks and shown insignificant levels of self-links. These findings concluded that *Ap* is a strong content consumer that is also illustrated in Figure 3.9 showing its all links as incoming links. There were no any significant results for strong content consumers in the news media in Facebook.

To summarize, the top 10 strong content consumers in the news media community are *Ap, Chron, Hollywoodreporter, Indianexpress, Timesofindia, Nytimes, Theatlantic, Thedailybeast, Usatoday*, and *Washingtonpost*. The content published by these news media shown to have the highest number of replicas that were previously posted by the remaining news media.

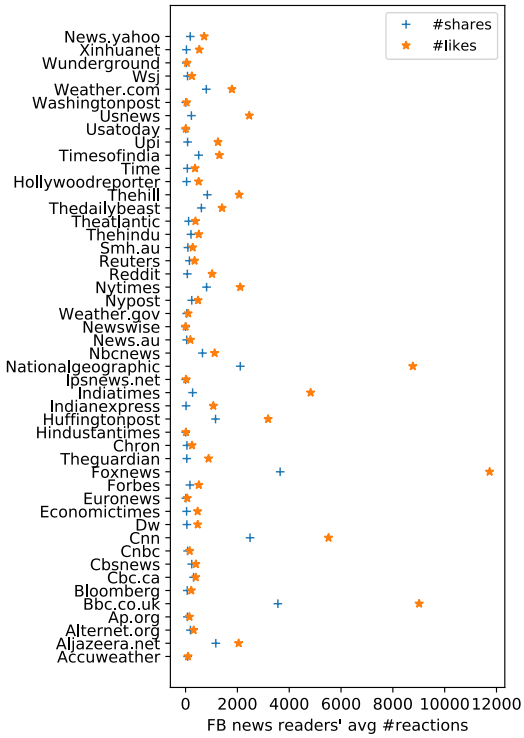
3.2.2.6 News Media Popularity

Marketers follow different strategies to measure type of reactions from their customers in social media. The simplest metric is to track number of likes or shares and also measure their audience growth rate of followers. This section aims to analyze news media in terms of reader reactions received on shared posts aiming to build a predictive model to increase future reader reactions of news media in Facebook and Twitter.

News reader reactions: To explore which news media content was widely interacted with news readers, we generated a dot-chart shown in Figure 3.10, which shows an average reader reactions per news media in both Facebook and Twitter. In addition, Figure 3.11 elucidates the distribution of reader reactions separately for Twitter and Facebook. As illustrated in Figure 3.11b, news readers on Facebook have exhibited a large number of likes (avg-1455) than number of shares (avg-454), and similarly in Twitter (Figure 3.11a),

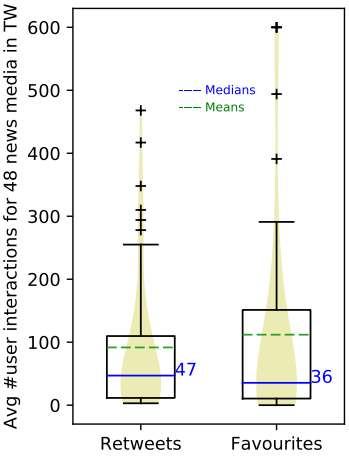


(a) Average number of news reader reactions in Twitter.

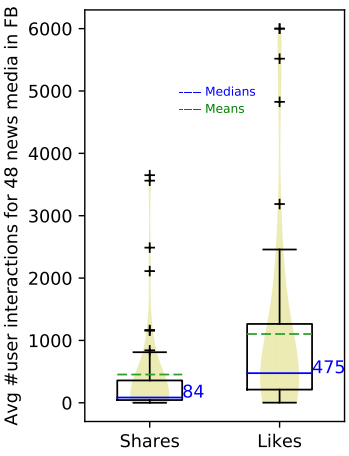


(b) Average number of news reader reactions in Facebook.

Figure 3.10 – Average number of news readers' reactions to the news items shared by 48 news media presence in Facebook and Twitter.



(a) Distribution of the news reader reactions in Twitter.



(b) Distribution of the news reader reactions in Facebook.

Figure 3.11 – Distribution of the news readers' interactions with the news items in Facebook and Twitter.

favorite count (avg-133) is higher than re-tweet count (avg-92). The comparisons of Figure 3.11b and Figure 3.11a indicates that Facebook news items were widely popular among news readers than tweets. The recent news analysis done by Pew [122] was also stated that Facebook is still the most commonly use news site by American adults.

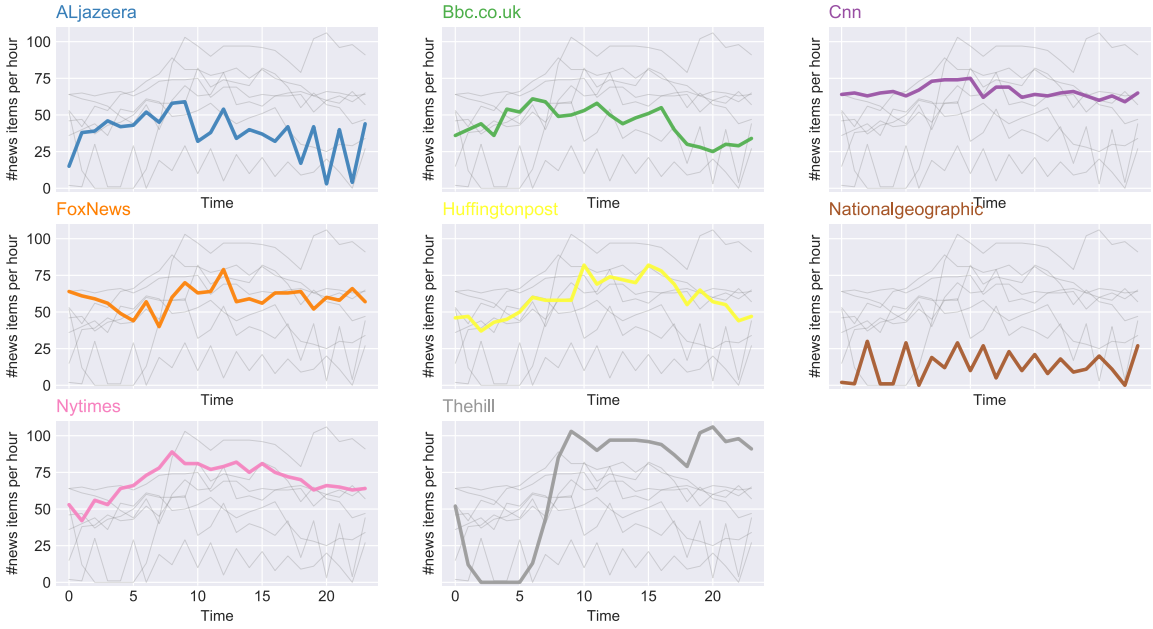
As shown in Figure 3.10, a set of news media has received much more attractions from readers on their published posts in Facebook and Twitter. The top 10 popular news media in Twitter in terms of both favorites and re-tweets counts were *FoxNews, Cnn, Nationalgeographic, Nytimes, Washingtonpost, Thehill, Ap, Huffingtonpost, Bbc.co.uk*, and *Aljazeera*. Among them, the highest re-tweets counts (about 1K) were exhibited by *FoxNews* and *Cnn*. The largest favorites count, approximately 0.5K, was also exhibited by *FoxNews* and *Cnn*. Whereas in Facebook, top 10 news media having large number of likes and shares were: *FoxNews, Bbc.co.uk, Cnn, Nationalgeographic, Aljazeera, Huffingtonpost, TheHill, Nytimes, Indiatimes*, and *Usatoday*. The *FoxNews* was shown to have 12K number of likes and 3.7K number of shares in Facebook and *Bbc.co.uk* was received the second largest number of likes and shares, 9K and 3.6K respectively.

To conclude here, we discovered most popular top 10 news media in Twitter and Facebook and identified 8 news media those who shared popular news items in both Facebook and Twitter as: *FoxNews, Bbc.co.uk Cnn, Nationalgeographic, Nytimes, Thehill, Huffingtonpost*, and *Aljazeera*.

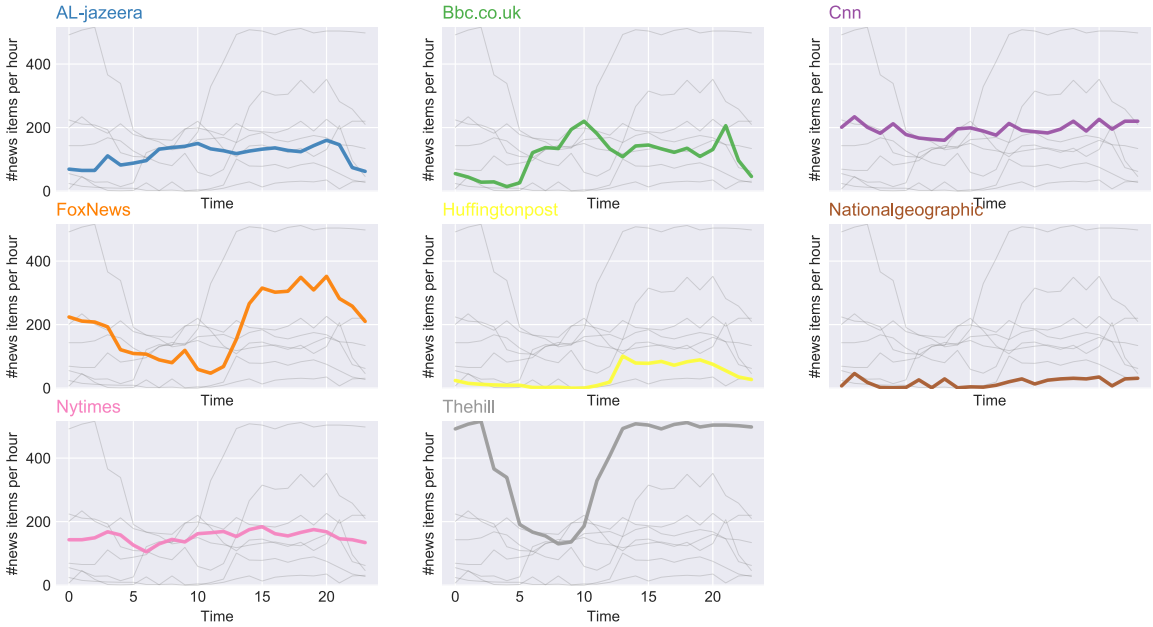
Circadian behaviors of popular news media: Any social media account has a unique time period to share contents (online circadian patterns). Posting time of a content is important to become popular among readers and to receive more reactions. Therefore, in this section we analyse online circadian behaviors of the most popular news media in Facebook and Twitter, identified in Section 3.2.2.6. This analysis helps to understand the best duration to post in Twitter and Facebook in-order to increase the possibility of reaching to a large audience.

In this section we try to understand the news sharing patters of news media and we focus only on those who are having very popular news items among news readers. As explained in the previous chapters, popularity of a news media is based on the reader reactions received on its news items and number of followers is not the only factor to become popular. For these analysis, we can consider online circadian patterns [126] of the popular news media to understand their publishing behaviors mainly considering the timestamps (in GMT standards).

Figure 3.12 elucidates the online circadian rhythms of the most popular 8 news media identified in Section 3.2.2.6. Interesting observation in both Facebook and Twitter circadian rhythms of these 8 news media is that, they tend to publish news items throughout the day except *Thehill* where it post news items during a specific time period in both social media.



(a) Circadian rhythm of popular news media in Facebook.



(b) Circadian rhythm of popular news media in Twitter.

Figure 3.12 – News sharing patterns of the most popular news media in Facebook and Twitter.

As shown in Figure 3.12a *FoxNews, Bbc, Cnn, Nytimes*, and *Huffingtonpost* were published throughout the day in Facebook and peak publishing time was approximately within 9am-12noon.In Twitter peak time of receiving highest attention from the readers was within 7pm-9pm as shown in Figure 3.12b. Interestingly, *Cnn* was shared news items in both Facebook and Twitter throughout the day in almost same frequency and its news items become very popular among news readers. To conclude, if news media publishes its news items throughout the day, then, news items will be popular among readers.

Reader reactions predictive model: This section focuses on exploring an existence of a correlation between news media content popularity with other attributes such as number of posts, self-originated, acquired and dispersed content. In this section, we focus only on the reader interactions of tweets as the highest number of posts were shared by the news media on Twitter than on Facebook.

Figure 3.13 exhibits a correlation graph (based on the Pearson correlation coefficient) for variables; number of posts, reader reactions and weights of SelfLink, DispersedLinks, AcquiredLinks of each news media in Twitter. The main observation is that, total number of Interactions are highly correlated with weights of selflinks and dispersed links and weights of acquired links are correlated with number of posts.

Next, we tried to explore whether a regression model can be implemented using different features mentioned above in order to predict news consumer reactions using regression analysis. The model examine three major attributes: number of posts, number of interactions ('In') and number of followers for predicting future reader reactions on news media in Twitter. As illustrated in Table 3.7 we built two linear regression models in-order to predict possible re-tweets and favorites counts of news media in future. We can observe that, among 3 response variables only the number of followers and 'In' have p-values closer to 0, and they are in the p-values range of 5%. Consequently, we can see a relationship between number of followers and 'In' with re-tweet and favorites counts, while number posts do not have much effect on the predictor variables.

These findings conclude that, in order to increase content popularity of a news media among readers they should perform more interactions in news community. As per the analysis of Figure 3.13, total number of interactions are correlated with the weights of self links and dispersed links and therefore we can conclude that news media have to share content that they themselves originated and they should distribute those content to other news media acting as content providers. Further, higher the number of followers in a news group larger the number of reader reactions received on the news items.

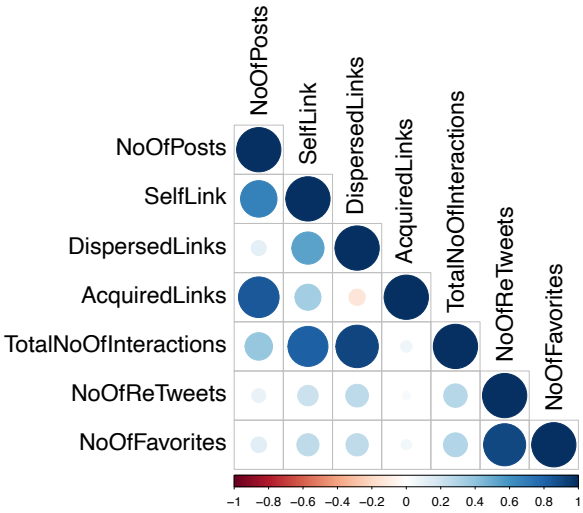


Figure 3.13 – This correlation plot shows the association between different attributes identified from the news media that are used in this study such as number of posts, SelfLink, DispersedLinks, AcquiredLinks and reader reactions

Table 3.7 – Regression models for re-tweets and favorites

		Estimate	Std. Error	t value	Pr(>t)
Re-tweets	**(Intercept)**	30.5	28.1	1.085	0.284
	#posts	-0.011	0.008	-1.351	0.184
	#followers	0.001	0.001	2.821	0.007**
	In	0.012	0.005	2.22	0.032*
	R-squared: : 0.221 and P-value: 0.011				
Favorites	**(Intercept)**	7.72	59.100	0.131	0.897
	#posts	-0.019	0.017	-1.067	0.292
	#followers	0.000	0.000	2.373	0.022*
	In	0.023	0.011	2.095	0.042*
	R-squared: 0.186 and P-value: 0.027				

Significant codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

3.2.2.7 Future works

The experiments conducted in this work are solely based on the features extracted from ConOrigina framework and other properties of the posts shared in Facebook and Twitter. One main component of the ConOrigina is to apply pre-processing techniques such as remove posts that are having word count length less than 3 and removing URLs, #tags, usernames from the post. Pre-processing the dataset helps to improve the efficiency of the model in terms of accuracy and also processing time. However, if we include attributes such as URL, #tags and @usernames in the dataset we can also explore the portion of content that are cited by others in Twitter. Even though we named those news media who share replica as content consumers, this is an important information to explore in detail. Therefore, as a future work we will analyse which news media is actually citing the replicated content.

Moreover, we will also explore dissemination patterns among international news agencies (Agence France Presse - AFP, Associated Press - AP, Reuters and United Press International - UPI) and other individual news media.

3.2.3 Experiment 3: Flaming Event Detection in Social Media

This experiment is conducted as an application to our improved sentiment classification algorithm.

3.2.3.1 Introduction

In modern Internet parlance, cyberbullying has become increasingly common, especially in the Internet communities and SNSs such as Facebook, Twitter, Instagram and YouTube. In this era, several new phenomena have appeared, leading to important debates and discussions and among them one hot topic is 'Flamings'. The Flaming can be considered as a serious issue on SNSs where many users express disagreements, insults or offensive words in the form of comments on a forum, blog or chat room intended to inflame emotions and sensibilities of others. These comments do not contribute any useful content to the discussion groups and instead attempt to wound another person socially and psychologically. These comments might be posted by genuine users, fraud or spam generated content [145]. Two popular examples of flaming include 1) 'Delete your account': a Clinton-Trump Twitter flame war[2] and 2) flame war between Donald Trump and Pope Francis on the pope's calling Trump 'disgraceful' for his immigration recommendations[3]. As a result of this high-level visibility, flaming has became an interesting topic among social researchers as they seek to understand the phenomena and explore the impact of these types of activities on targets.

One use-case we target in this experiment is news media in Facebook as many American adults consume news on social media and majority of them are commonly use Facebook [122]. In addition we can observe that the number of fans in news media Facebook pages[4] are considerably higher and therefore receive many feedbacks from the readers. News media tend to publish news items of interest to more diversified and varied readers, and therefore many news readers interact with news items daily via commenting, sharing and reacting. A set of news media is relying on the content of most popular news media as content consumers [118], [119] and so, it is important to understand these flaming types of events in the news media domain. Thus, our main objective is to explore news items' flaming events in terms of negative feedback with insults and other offensive words. The existence of flamings on news items may reduce number of followers, or sometimes these types of

[2]https://www.sbs.com.au/news/delete-your-account-clinton-trump-in-twitter-flame-war
[3]https://www.lifewire.com/what-is-flaming-2483253
[4]https://fanpagelist.com/category/news/

posts can go viral and increase the number of followers. Therefore detecting flamings and identifying the topics that the community strongly disagree with is a useful conception for news media.

Sentiment polarity prediction is one of the main ways of detecting flaming events in SNS [146], [147]. Many advancements have been made to the sentiment classification methods to date. However, these methods are domain-specific and therefore their results are strongly biased on the words used in that domain. As a result, we build a word embedding-based multiclass (5-classes) sentiment classification deep Neural Network (NN) based approach by focusing on Facebook news items and variable length user feedbacks that can be modified and applied in any other social media category other than news. In addition, we use an improved lexicon-based sentiment classification method to generate a true labeling list with which to train the deep NN model. The flaming effect analyses will be done based on the comments classified as Negative and Very Negative. A flaming event takes place when many users give negative feedback, and so a post with a large number of negative comments received within a short time will possibly a flaming event. We will also explore what types of topics were mainly affected by flamings that are published by BBCNews, CNN and FoxNews.

Different contributions we focus with this experiment are:
1) propose a word embedding-based sentiment prediction deep NN model focused on Facebook comments.
2) explore which word embedding method (Word2Vec or FastText) works better on Facebook comments.
3) identify flaming posts on BBCNews, CNN and FoxNews Facebook pages published in February 2018.
4) identify flaming posts' associated topics.

3.2.3.2 Literature Survey

Researchers have investigated flamings on YouTube [148], email threads [149] and comments received on Twitter and Facebook [146], [147], news sites and news channels [150]. These works show that flaming events can be appeared in any online platforms especially in the SNSs as any user can comment on a public content. Sentiment polarity prediction is one of the main ways of detecting flaming events in SNSs [146], [147]. Sentiment analysis of social media content has become more and more popular, as it can be used for mining opinions on services, products, companies, etc. and these models can be implemented as supervised, unsupervised or semi-supervised [151] methods. However, with the increase of user-generated content in SNSs it is not effective to apply lexicon-based unsupervised methods, and therefore supervised methods can be automated to detect polarity. Apart from

that, analyses on sentiment classification are aspect-based and domain-specific. Therefore, model needs to train with domain specific data to achieve better accuracy when building a sentiment prediction model. Recently, it has been widely acknowledged that deep learning-based representation models have achieved great success in text sentiment classification tasks compared to traditional machine learning models [152]. Furthermore, in recent works, word embedding-based method are applied for sentiment classification. A few have used Word2Vec embedding [153], [154]. Deep learning has emerged as a powerful machine learning technique and is also popularly used in sentiment analysis in recent years [155], [156]. Wang et al. proposed a CNN-RNN architecture [157] to analyze the sentiment of short text while some other studies were used methods based on CNN [158] and RNN [159]. Their experimental results shown that the proposed method outperforms lexicon-based, regression-based, and obtained an obvious improvement upon the state-of-the-art. In addition, many research works considered only the binary sentiment classification and few studies have used multi-class sentiment classification producing promising results [160].

3.2.3.3 Dataset description

News media flaming events can be detected by identifying negative comments received on shared news items in SNSs. Therefore, sentiment prediction models can be adapted to cluster senses of the user feedback in order to explore the existence of flaming events in news media in SNS using a rule based technique. Many previous sentiment analysis works have used Twitter as the SNS to analyze and build their sentiment prediction models. As Facebook employ different content properties such as permitting to share variable length posts and comments, unlike in Twitter, we build our methodology by proposing a sentiment prediction model focusing on news items shared on Facebook. Most efficient sentiment analysis algorithms are supervised learning which requires sufficient amount of training data. Hence, we introduce a deep-NN based supervised sentiment prediction model that require considerable number of true labels to train. The labeled dataset is generated using an unsupervised model. Therefore, this study targets on top three news media (BBCNews, CNN and FoxNews) in Facebook having the highest number of fans as a usecase to explore flamings. By respecting the ethical aspects of data collection process, we collected only the texts of public posts, comments and timestamps of these Facebook accounts and do not collected neither sensitive data nor personal data. The analyses are mainly based on this dataset and a brief description of the dataset is shown in Table 3.8. The training and experimental datasets are separated after pre-processing the original news items and user feedback.

We applied basic pre-processing techniques such as replace URL with a space, remove user mentions, hashtags, retweets, special characters and multiple spaces, removed multiple

Table 3.8 – Statistics of public posts obtained from BBC, CNN, and FoxNews Facebook pages during February 2018. (PP represents the abbreviation for Pre-Processing)

News media	#Fans	Number of comments of			
		300 posts Before PP	300 posts After PP	100 PPposts Training	200 PPposts Experiment
BBCNews	46.2M	398,453	312,881	107,874	205,007
CNN	29.9M	595,268	312,881	217,386	288,606
FoxNews	16.3M	1,162,734	312,881	280,342	773,439

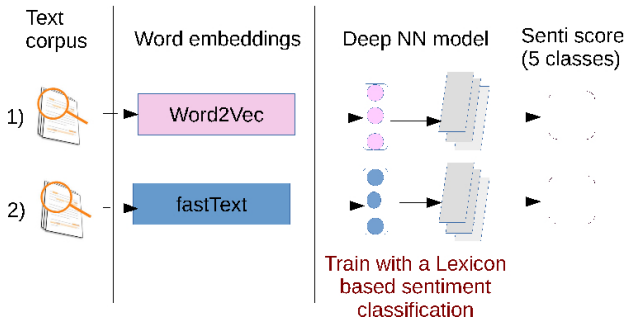


Figure 3.14 – The architecture of the multiclass sentiment prediction using word embedding

letters from a word ('haaappy' will be replaced by 'happy') ,merge characters of a word written with spaces or dots in between each character ('h a p p y', 'h.a.p.p.y' will be reformatted to happy). As the objective of this work is to identify the sentiment of SNSs content, emojis impact a lot on the overall sentiment value. Therefore, we included all the positive and negative emojis in the text without ignoring. We also performed stemming on each word in the sentence to generate the stem. Lemmatization is not applied as lemmatizers have to search through a corpus while stemmers do basic string operations. Also, stemming can work with unknown words while lemmatizer do not recognize them. We feed to the neutral network separately the datasets after being lemmatized and stemmed, and the accuracy of the model was very low with the lemmatized dataset. We do not remove stopwords in sentences while pre-processing.

Considerable amount of comments have been removed after preprocessing and the main reason was that the majority of those comments were URLs. We can observe from the Table 3.8 that news items published by FoxNews has received a higher number of comments, while having less number of fans, than BBCNews and CNN. We use comments of 100 pre-processed posts from each news media as a training dataset to train the NN model and comments of the remaining 200 news posts from each news media as the experimental dataset to explore flaming events.

3.2.3.4 Methodology and Proposed Solution

Proposed sentiment analysis architecture: The architecture of our variable length text sentiment prediction model is shown in Figure 3.14. At first, as depicted in Figure 3.14, we build a lexicon-based approach to generate the training dataset for our deep NN model. Many previous studies have tried to employ word-embedding methods for machine leaning or deep learning-based sentiment classification [153], [154], [156]. In our prediction model we experiment with two different word embedding methods: Word2Vec and FastText. Word2Vec is widely used for both shallow and deep neural networks that was developed by Google in 2013 [161]. In the Word2Vec model, words with similar meaning are mapped to a nearby vector space, thereby making it simple to explore semantically similar words. As the Word2Vec model is based on a predefined dictionary, one shortcoming of this representation is that rare words may not be mapped with other vectors. However, FastText was developed by Facebook [162] in 2016 using the n-gram representation of each word in a sentence and therefore helps to detect misspelled words and slang across different languages. In the experiments we will explore which word embedding method is more suitable for sentiment classification of the Facebook content. The deep NN model shown in Figure 3.14 uses the best performed word embedding method to predict the sentiment of a sentence based on a multiclass (5 classes) classification technique, where five different classes can be defined as; Very Positive(4), Positive(3), Neutral(2), Negative(1) and Very negative(0).

Methodology for generating true labels: The deep NN sentiment prediction model need to use a set of true labels to train the model. We introduce an improved lexicon-based approach to identify the sentiment of Facebook content and then use this model to build the training corpus for a deep NN model to automatically classify the sentiments. In this approach we can customize sentiment labels to be 2-class (positive, negative), 3-class (positive, neutral, negative), or 5-class(very positive, positive, neutral, negative, very negative) depending on the requirement. This section explains this unsupervised lexicons-based sentiment classification approach.

The consideration of negators, adverbs and model verbs associated with words is important for sentiment classification. Therefore it is worth to identify important sentiment of words and word phrases separately including various negators, models, and degree adverbs, as well as their combinations. Kiritchenko et al. [163] created a lexicon dataset, Sentiment Composition Lexicon for Negators, Models, and Degree Adverbs (SCL-NMA) and it includes 3207 phrases with related sentiment scores where each phrase contains at most 4 words. Sentiment value of a phrase ranges from -1 to 1 where phrases with negative meanings have assigned negative values and phrases with positive meanings have assigned positive values. A sentence will be assigned a sentiment value closer to 1 when the its

meaning is more positive and similarly, sentences having very negative meanings will be assigned score very closer to -1.

We used an statistical based approach to identify the polarity of a sentence in English. A sentence may have one or many lexicons found in the SCL-NMA dataset and also both negative and positive lexicons. Hence, we explored all the lexicons in the sentence to classify its overall sentiment. This approach is very useful to identify the sentiment of variable length sentences and following equation shows the prediction of sentiment of a new sentence. Assume that SCL-NMA dataset consists of lexicons L1, L2,...,Lx.

$$SentiScore = \frac{[\sum_{n=1}^{N} Ln] + [C + S + E]}{[N + C + S + E]} \tag{3.1}$$

Where N is the number of lexicons in the sentence. Letters C, S, and E are constant and set to be 1, -1 or 0 depending on the lexicon polarity. These constant values are added to the SentiScore formula based on the existence of different properties of the lexicons in a sentence as explained below.

The overall sentiment of a sentence is increased when it has lexicons (found in the SCL-NMA dataset) with capital letters. Therefore, we identify those lexicons in capital letters and use these details in Equation 3.1, the letter C, to evaluate the overall sentiment score.

$$C = \sum capital\_lexicons \tag{3.2}$$

where *capital_lexicons=1*, if the lexicons' in capital letters has a positive sentiment value, *capital_lexicons=-1*, if the lexicons' in capital letters has a negative sentiment value and *capital_lexicons=0*, if no capitalized lexicons in the sentence.

Emojis in a sentence are affecting more on the sentiment score. Thus, E in Equation 3.1 refers to all emojis in the sentence those that helps to increase the strength of the sentiment.

$$E = \sum emojis \tag{3.3}$$

where *emojis=1* for a positive emoji, *emojis=-1* for a negative emoji and *emojis=0* if no emojis presence in a sentence.

People use exclamation marks to stress words and this is an important property when evaluating sentiments. Thus, we observe the presence of exclamation marks in a sentence and this feature is presented in Equation 3.1 as S.

$$S = exclamation\_mark \tag{3.4}$$

where *exclamation_mark=1* if a positive lexicon is attached to exclamation mark, *exclamation_mark=-1* if a negative lexicon is attached to exclamation mark, and *exclamation_mark=0* if no any exclamation mark is attached to lexicons.

Table 3.9 – Confusion matrix of our lexicon-based sentiment classification method (macro precision-60.66%, macro recall-62.01%, F1 score - 61.31%)

		Predicted			
		Positive	**Negative**	**Neutral**	**Precision**
Actual	**Positive**	128	19	35	70.33%
	Negative	57	83	37	46.89%
	Neutral	37	12	90	64.75%
Recall		57.66%	72.81%	55.56%	

Table 3.10 – Confusion matrix of Vadar sentiment classification (macro precision-37.85%, macro recall-51.41%, F1 score - 43.59%)

		Predicted			
		Positive	**Negative**	**Neutral**	**Precision**
Actual	**Positive**	24	12	156	12.5%
	Negative	0	15	162	8.46%
	Neutral	1	1	137	92.57%
Recall		70.56%	53.57%	30.11%	

The algorithm consider n-grams as the features, where n ranges from 1 to 3 and generate features separately for lexicon dataset and sentences that need to identify its sentiments. Equation 3.1 helps to identify the sentiment of any sentence written in English regardless of the length of the sentence. This approach helps to generate a labelling dataset from user feedback received on news items shared on Facebook in order to train our deep NN model.

Comparing unsupervised true lable generation method with a baseline method: VADAR lexicon and rule-based sentiment analysis tool [164], which is widely used for classifying social media content, is considered as the baseline method to compare our unsupervised true label generation model. For a given sentence, VADAR returns its sense from 3 classes: positive, negative or neutral. Therefore, in order to compare our algorithm with VADAR, our model is customized to classify sentence sentiment in to the same 3 classes.

We use a manually annotated dataset having 498 sentences with its sentiment values: 177 negative tweets, 182 positive tweets and 139 neutral tweets published by the Stanford University [165]. This dataset is applied on both VADAR and our unsupervised sentiment classification technique and their confusion matrices are shown in Table 3.10 and Table 3.9, respectively. The precision, recall and F1 score parameters of our approach is much higher than VADAR for the classification of positive and negative sentiments as detailed in Table 3.10 and Table 3.9. The results is proven that our lexicon based approach can classify the sentiment label of a sentence much better than VADAR. However, we can still improve the accuracy of our model by increasing the lexicon dataset with additional lexicons.

This approach is unique as our training dataset for the deep NN can be generated automatically, as opposed to having manual annotate Facebook posts and comments. However,

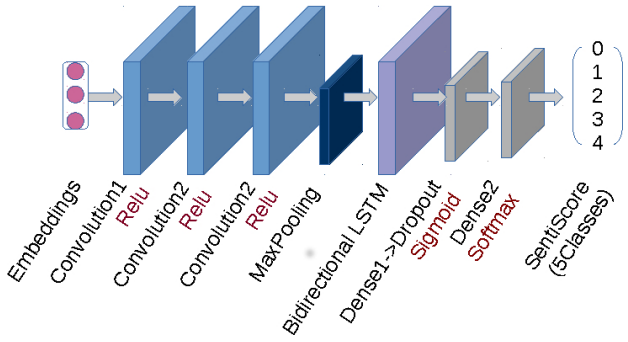


Figure 3.15 – The deep model for extracting sentiment scores

there is still a room to improve the classification accuracy of our supervised model for making better predictions. Moreover, this model is introduced as a multiclass sentiment classification approach rather than classifying only into 2 or 3 classes. In this model, we assigned a sentiment label for each sentence from five classes as follows.

$$Prediction = \begin{cases} VeryPositive, IF(sentiScore => 0.5) \\ Positive, IF(0.5 > sentiScore > 0) \\ Neutral, IF(sentiScore = 0) \\ Negative, IF(0 > sentiScore > -0.5) \\ VeryNegative, IF(sentiScore <= -0.5) \end{cases} \tag{3.5}$$

The baseline value for distinguishing Very Positive from Positive and Very Negative from Negative classes is set to be 0.5 and -0.5 respectively.

Methodology for an automatic sentiment prediction model:
Since deep learning-based sentiment classification models are better than the traditional machine learning models [152] we try to adapt one deep learning-based model in this work to predict sentiments of Facebook content. These deep models can learn text representation from original data that can capture relations between contexts words in a scalable manner.

Recent research on text sentiment classification used convolution operations using n-gram features. However, the convolution neural network (CNN) completely ignores the sequence information of the text while paying attention to the local features of a sentence. On the other hand, long short-term memory (LSTM) networks are good at learning sequential correlation in the text by using an adaptive gating mechanism but lost the ability to learn local features of the text. Therefore, to involve both sequence correlation information and local features, it is important to explore an effective combination strategy that takes advantage of the CNN and the LSTM network. Hence, we experiment with several different architectural methods such as:

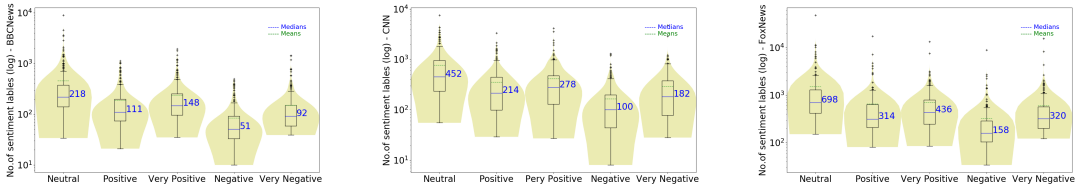
i) CNN-LSTM-Word2Vec : pre-trained Word2Vec vectors,
ii) CNN-LSTM-Word2Vec : custom Word2Vec model train with our own data,
iii) CNN-LSTM-Fasttext : pre-trained Fasttext vectors,
iv) CNN-LSTM-Fasttext : custom Fasttext model train with our own data, and
v) Adjusting the number of CNN layers and dropout layers, and maxpooling layer.

Models are trained separately with the custom Word2Vec and FastText embedding methods using 315K total number of comments (experimental dataset in Table 3.8) with their sentiment labels generated from our lexicon-based method. The training dataset consist of 42.3%-neutral, 20.5%-very positive, 16.6%-positive, 12.9%-very negative and 7.7%-negative labels. And also these models are trained with both pre-trained word embedding. Most promising results with high accuracy is given to the architecture shown in Figure 3.15 and achieved high accuracy with our own domain specific word embedding model rather than using pre-trained embedding.

As shown in Figure 3.15, model was implemented with three CNN layers and a Bi-LSTM layer. The inputs of the NN model uses 1D convolutions. One dropout layer was added within two dense layers (hidden layers) close to the output layer, and another dropout layer on the Bi-LSTM layer. The model compilation is done by adjusting three parameters: loss, optimizer and metrics. The Adam [166] optimizer was used in our model to adjust the learning rate throughout the training and the learning rate was fixed to be very small (0.0001) leading to obtain more accurate results. The categorical_crossentropy loss function was used in our implementation. This model uses three activation functions: Relu- for CNN layers, Softmax- for the output layer and Sigmoid- for the dense layer. The method of dealing with variable length sentences is that, at first we need to set a maximum number of token values (30 in this research work) and then if a comment has more than 30 tokens, we divided it into sub-sentences and evaluate the sentiment of each sub-sentence separately. Finally, the sentiment score for the entire sentence is the average sentiment scores of sub-sentences.

At first, deep NN model is trained for 40 epochs and evaluated the accuracy and validation loss for each iteration separately for Word2Vec and FastText. In the Word2Vec model, we analyzed that at the 13th epoch, the performance of the training dataset continue to decrease than the validation dataset, indicating an over-fitting. Therefore, the best-fit for this model is to training with 12 epoch as the model has good skill on both the training dataset and unseen test dataset. The accuracy of the model is identified as 85%. Similarly, when the FastText model used in the deep NN model, it shows the best fit after 15th epoch, in which training and validation loss is almost equal and with 78% accuracy. In both scenarios, total number of parameters trained by the NN model is 3,050,053.

The results manifested that, Word2Vec model out performs the FastText model when



(a) BBCNews (neu-41%, pos-38%, neg-21%)

(b) CNN (neu-39%, pos-38%, neg-23%)

(c) FoxNews (neu-40%, pos-36%, neg-24%).

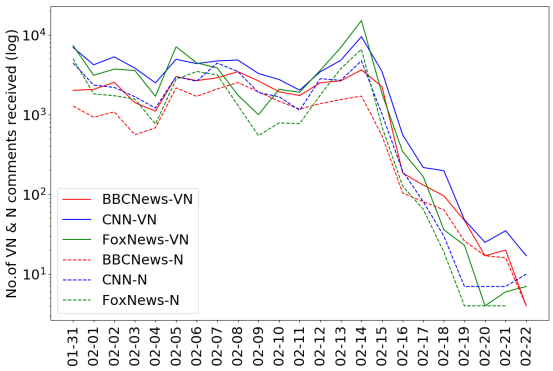
Figure 3.16 – Five class sentiment value distribution of 200 random posts shared by BBCNews, CNN and FoxNews in Facebook during February 2018.

applying for multiclass sentiment classification of Facebook content. As a result, we will use our generated models with Word2Vec embedding to explore the existence of any flaming events in news media in Facebook.

In social media, relatively large number of critical comments can be directed at individuals, companies, brands and etc. and this behavior is called as flaming. One way of exploring flaming behavior is to identify whether a post received higher number of very negative comments within a short time. This section explains, how the implemented deep NN sentiment prediction model can be used to identify the existence of flaming or similar kind of event in news media in Facebook.

Experimental dataset: In order to analyze flaming events and their existence in news media in Facebook, we randomly selected comments from 200 posts (Table 3.8 and Figure 3.16) shared by BBCNews, CNN and FoxNews in February 2018 as our experimental dataset. First, we try to explore sentiment labels for each comment in the experimental dataset using our deep NN model which is trained with the Word2Vec model as Word2Vec model given a high accuracy than the FastText model.

Figure 3.16 shows the statistics of the sentiment prediction of the comments belongs to 600 posts in our experimental dataset (200 posts from each BBCNews, CNN and FoxNews). As indicated in Figure 3.16, number of Neutral comments received on these posts are much higher than the other classes. The second most number of sentiment predictions of comments are belongs to the Positive class, followed by Very Positive and Very Negative. The least number of predicted sentiment values are from the Negative class. In general, number of Neutral comments received on the posts are always higher than the total number of positive comments (Very Positive + Positive) and total number of negative comments (Very Negative + Negative). We observed that the text classified into a Neutral class have a large number of stop-words, names, URLs, and single words without indication of any feelings.



(a) No.of N/VN comments received on posts per day

(b) No.of VN comments received on posts shared in 14th February 2018.

Figure 3.17 – The distribution of Negative (N) and Very Negative (VN) comments received on posts shared by BBCNews, CNN and FoxNews in February 2018.

3.2.3.5 Analysis of Flaming Events

Flaming events in news media - time-series approach: Receiving avalanches of negative emotions or flood of low happiness comments from users within short period of time tend to generate flaming events and a time-series approach can be one of the optimal ways to explore it. Figure 3.17 exhibits the time-series distribution of received Negative and Very Negative comments in our experimental dataset. Figure 3.17a exhibits negative comments received per day during February 2018. As shown in Figure 3.17a, number of Very Negative comments is always higher than the number of Negative comments and therefore we focus only on Very Negative comments for further analysis. We can observe that, on 14th February 2018, a flood of negative comments have received on posts shared by all three news media compared to the other days. Apart from that, 31st January, 2nd and 5th February exhibited another flaming type of distributions showing few other spikes of number of Very Negative comments.

As shown in Figure 3.17a, the main flaming type of event taken place in 14th February. Therefore, to identify more information about this event the consideration of the posts shared during one day before and after 14th February is also important. Figure 3.17b shows this distribution in terms of the Very Negative comments received during these 3 days. We can observe that all three news media have shared the flaming posts in 14th February and received a large number of Very Negative comments but, FoxNews has shown a flaming type of event on 13th February where some of the comments on these posts have received on 14th February as well. One main inspection from these figures indicate our hypothesis that the flaming type of posts received Very Negative comments within short duration of time (approximately 2-3hrs) after posting news.

Figure 3.17a and 3.17b indicate varied values of number of comments received on 14th February. We explored that on 14th February, BBCNews, CNN, FoxNews have published only 14.5% posts, 19% posts and 16.5% posts respectively out of 200 posts. Next we identified how many posts have received a higher number of negative comments than the other types of comments. The statistics shown that only 26.6% posts from BBCNews, 34.2% from CNN and 30.3% from FoxNews have received more negative comments than the other types of comments on 14th February. Hence, as per this analysis, only a set of posts contributed to the flaming event on 14th February 2018 while other comments have received on the posts shared by the other days. Next section explains more about flaming event posts and their discussed topics based on one statistical approach.

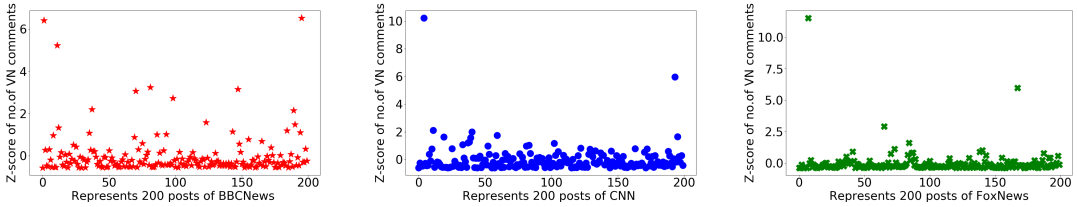
Flaming events in news media - statistical approach: The analysis of comments based on the time-series data has proven an existence of flaming event in BBCNews, CNN and FoxNews. Identification of types of the posts of these flaming events are useful and therefore in this section we will explore more details about the posts that attracts flaming comments.

At first, we try to identify these flaming posts with the use of one statistical approach that can predict these types of behaviors considering an outliers prediction method where outliers in this scenario are the posts those that received many insults or offensive comments. The standard score (z-score) is one outlier prediction method which allows to identify whether a particular value is equal to the mean, below the mean or above the mean. We calculate z-score for a distribution of number of Very Negative comments received on each posts within February 2018 using following equation where μ and σ represents mean and standard deviation, respectively. For each post we assign 'x', the number of Very Negative comments.

$$z = \frac{x - \mu}{\sigma}$$

Figure 3.18 exhibits the distribution of z-score values of Very Negative comments received on posts in our experimental dataset. As shown in Figure 3.18a, BBCNews has three outliers having z-score value greater than 5. Figure 3.18b exhibits two outliers for CNN with the z-score above 6 and as shown in Figure 3.18c, FoxNews has two outliers having z-score greater than 6. Next, we will analyze the popularity of these 7 outliers posts and their discussed topics.

Table 3.11 contains the information on above identified 7 outliers including published date, total number of reactions received, number of comments received, number of Very Negative comments (VN) and the discussed topics. We observed that, these posts are widely popular and have received enormous amount of reader reactions (number of reactions) compared to the other posts in the dataset. We also observed that these posts have received



(a) BBCNews (b) CNN (c) FoxNews

Figure 3.18 – The distribution of z-score values of number of Very Negative (VN) comments received per post (200 posts in total).

Table 3.11 – Statistics of the flaming types of posts shared by BBCNews, CNN and FoxNews (VN stands for Very Negative).

News media	Date	No. Of Reactions	No.Of VN Comments	Topics
BBCNews	2018-02-01	13590	4110 (44.28%)	Palestinian teenager slapped an Israeli soldier.
	2018-02-02	33110	3946 (37.56%)	A father has tried to attack a doctor who abused his three daughter.
	2018-02-14	28718	2614 (63.54%)	Shooting at a Florida high school.
CNN	2018-02-02	113632	19598 (30.99%)	A father has tried to attack a doctor who abused his three daughter.
	2018-02-14	39611	4670 (73.17%)	Shooting at a Florida high school.
FoxNews	2018-01-31	86717	130185 (20.61%)	House Democratic Leader Nancy Pelosi and other House Democratic leaders hold a news conference.
	2018-02-13	72765	35619 (29.41%)	The women of 'The View' took a shot a Vice President Mike Pence's Christian faith on Tuesday, mocking the former governor of Indiana for talking to Jesus and even calling it a 'mental illness'.

huge amount of comments and more of them are Very Negative (number of comments). In addition, all 7 posts have received more than 20% of the comments as Very Negative. An interesting observation of this analysis is to explore the types of topics discussed by each news media in these outlier posts.

Table 3.11 exhibits that BBCNews and CNN have posted similar topics on
i) 2nd February - about a father was trying to kill a doctor who abused his daughters and
ii) 14th February - about an incident of shooting at a Florida high school.

In addition, BBCNews has an extra flaming event on
iii) 1st February - about one Palestinian teenager slapping an Israeli soldier.

On the other hand, two flaming posts published by FoxNews are different to what BBCNews and CNN have shared on
iv) 31st January - type of a political discussion and
v) 13th February - related to a religious discussion.

In summary, we identified only 5 posts associated to the 7 outliers detected in the previous section. Therefore, we can conclude that, as exhibited in Figure 3.17 31st January and 2nd, 13th, 14th February exhibited flaming types of news posts with the analysis of a statistical method. The topics discussed on these flaming posts are mainly very sensitive information, political related details or related to religious matters. We can deduce from these information that, people try to send more aggressive and negative comments on sensitive news items.

3.2.3.6 Future works

These results in this experiment conclude that we can detect flaming events in news media in Facebook using a sentiment detection approach, and we can apply this analysis to other types of categories in Facebook such as celebrities, politicians, etc. Moreover, if the SCL-NMA dataset can be improved by adding more hateful and insult wordings with their sentiment scores, we can detect flaming events with higher accuracy. Our deep NN model can also be used to adapt to other use-cases, as it is a general model to detect the sentiments of textual content from five different classes. The proposed model can be enhanced to identify flaming wars in SNSs as the model we developed can be automated and behave as a semi-supervised approach. The comments received on flaming posts might be shared by the real users or might be Spam content [121]. Therefore, as future works, it is important to identify these patterns of posting flaming events based on commentator's behavioral properties or other parameters to detect who was the actual content publisher (i.e. human or bot).

One most important feature that we need to improve in the proposed deep neural net-

work is its training dataset. Our main goal was to propose a technique to generate labeled dataset automatically without any human interventions. As the initial approach we have used lexicon based approach to generate automatic labeling with improved classification results. However, we can merge several other techniques to improve th quality of automatic labeling without using only the lexical base technique. We can use clustering based techniques which does not require any lexical dataset to generate the label, instead it cluster unlabeled corpus into 3 clusters, namely positive, negative and neutral. We can focus on POS tags and each term can be weighted using TF-IDF at features to cluster sentences. In this scenario, we can use Expectation and Maximization (EM) framework [167] for finding maximum likelihood estimates iteratively. First, we train our dataset with the lexical based approach and then use trained classifier to assigned probability weights for each sentence. We can repeat the same process by merging with the original dataset and newly generated labeled dataset with sentiment scores. At each iteration, the sentiment score of each sentence will be improved since the likelihood of the parameters is improved until no more any changes are there. However, there is still more room to improve the generation of labeled dataset in this approach with various techniques as these approaches are really useful when identifying sentiment scores of unpopular languages in the world.

3.2.4 Experiment 4: Changed Agenda Topic Detection (CAT) in a Realtime Meeting

The aim of this experiment is to use topic detention and segmentation algorithms to explore topics discussing in a realtime meeting.

This CAT framework and the complete set of API is designed and proposed for a direct project collaborated with Orange[5]. This project is aimed at building meeting's mind map by identifying topics that are discussing in realtime. The main objective of the CAT framework is to detect topic changes during the meeting in realtime and more generally to provide insights about information that could be interesting to detect if we place the meeting into his global context (project, people, organization etc.). After implementing our proposed framework, we tested and evaluated CAT module with their internal dataset (both in French and English), and then integrated to their internal platform. The proposed CAT module is integrated to their internal system as a REST API with their and obtained satisfactory results.

This experiment is deviated from the main objective of my thesis especially in terms of the dataset. As explained above, the objective of this work is mainly an application that can fulfil a requirement in industry. However, in order to begin with the implementation, first we conducted a comprehensive research on what type of algorithms and approaches are

[5]https://www.orange.com/en

applicable to this scenario and then based on the identified approaches we implemented the framework. Even though the dataset is used in this experiment is different to social media data, this is also a research related to NLP. Hence, we applied set of text pre-processing techniques and data preparation approaches as we are dealing with textual content that are spoken by human where, these types of dataset consists of many slang and spoken idioms.

3.2.4.1 Introduction

Spoken multi-party meetings are an important activity in organizations and different kind of physical and remote meetings occur frequently in business daily life. During a meeting, massive amounts of data must be recorded in text or audio format. The quality of records affects the organization process because the decisions made in meetings usually frame the rules for future work. To obtain meeting records in high quality, human effort is required when annotating transcripts and summaries. Most meetings discuss many topics instead of a single one, and the participants may have different views for each topic. Meeting participants specially group members who join a meeting late or attend from a different location are likely to want to know which topics were discussed in a particular meeting, as well as have access to the discussion on particular topics in which they are interested. Of course, this requires both identification of the topics discussed, and segmentation into the periods of topically related discussion. Thus, it is convenient to understand a meeting's progress by segmenting text and retrieving topics [168]. Spoken multi-party meetings pose some difficult problems such as:
i) Neither the segmentation nor the discussed topics can be taken as given.
ii) The discourse is by nature less tidily structured and less restricted in domain.
iii) Speech recognition results have unavoidably high levels of error due to the noisy multi speaker environment [169]. With few textual and structural features, detecting topics and providing summarization is a significant challenge.

One of the main part of a meeting is indexing meetings for the subject matters that users find salient, including topic boundaries, decisions, intense discussions, or places in which a specific person or subject was mentioned. This requires making manuscripts from meetings at first. Meeting manuscripts are significantly different from written text and other audio data. In meeting transcripts, information could be hidden and difficult to identify the important data [170]. Meeting transcripts do not follow the rigid pattern of grammar and punctuation formatting associated with written text. The members play several roles and discuss various topics in the meeting transcripts and there are many dis-fluencies in spontaneous speech. For instance, meeting transcripts may deal with the participation of various members where the deliberations are not well organized and the speech is impulsive, with unformed sentences. Each individual may be speaking with different accent, different

languages, pronouncing words and slang differently in the meetings [171].

Therefore, one challenging problem is to detect topic shift during a meeting that lead us to create the Mind Map of the meeting. The main idea in this experiment is to identify the topics of a discussion and later detect when the participants of the meeting changed the topic in the discussion. Topics are probabilistic clusters of words in text corpora that are semantically related to each other and they are represented as a set of descriptive and collocated terms. To find these hidden topical patterns of words in a text, topic modelling can be used. Topic modelling is a type of statistical model that finds latent variables or hidden structures in the data by applying clustering approach as an unsupervised machine learning technique.

With fast development of recording and storage techniques, large amount of meeting transcripts are becoming readily available resources rather than structured text form nowadays. Therefore, taking into account these meeting records and try to develop techniques to identify topic change, opinion detection, action detection, and decision detection during a meeting is a positive challenge for organizations and management systems [169]. Identifying the topics discussed in a meeting is very important to generate meeting minutes automatically and important to build the mind map of the meeting in real-time also. The discussed topics in a meeting can be extracted from meeting transcripts. However, meeting conversations are intrinsically different from well-written text, as meetings may not be well organized and most utterances have low density of salient content. In this experiment we use topic modelling approaches to detect the topic shift in an ongoing meeting. In the following sections we will describe some meeting mind map generation tools used for textual data or manuscripts. Then, the proposed methodology and dataset considered in this work will be presented. Finally, the results achieved from the most common methods of topic modelling and its performances will be presented with possible future research directions on this research question.

3.2.4.2 Literature review

Here there are some recent research efforts for topic detection and summarization of meeting corpus. Some of them just consider words of the meeting and apply purely text-based approaches to extract the meeting topic/summary, while others alter the text-based algorithms to incorporate additional information such as speakers. In [172] a new topic extraction method is proposed. It extracts topics according to the flow of conversation. Although, this method extracts the appropriate topic words according to their importance in the conversation in a time series based on text data from the meeting (meeting transcripts) since the importance of topics in a meeting changes in a time series. In [173], an unsupervised topic modelling is proposed to infer a set of semantically coherent topics while providing

a segmentation into topically coherent segments over multi-party discourse transcripts. It uses a generative unsupervised approach to model the text as being generated by a sequence of mixtures of underlying topics. For topic segmentation, this method models each utterance as being generated from a particular distribution over topics, where each topic is a probability distribution over words. The utterances are ordered sequentially. In [169], a framework for detecting discussion topics and summarizing meeting transcripts is proposed. It includes three parts: speech recognition, topic detection, and extractive summarization. In fact, an application to help the users find topics and obtain summarization of meeting contents without any extra effort was developed. This application uses the Bluemix speech recognizer (IBM Watson) to obtain speech transcripts. It then combines latent Dirichlet allocation algorithm with the speech script of meetings to detect boundaries between different topics and evaluate the topics in each segment. To select the most recently used terms, this research adopts Wikipedia as a training corpus to generate a dictionary through the LDA model. This corpus contains numerous articles which include terms and their related tokens. Finally, TextTeaser [174], an open API based on a feature-based approach, is used to summarize the speech transcripts. To measure the performance of the model, the AMI Meeting Corpus, a multi-modal data set consisting of 100 hours of meetings recorded by the AMI project, is used as the data source. This work does not incorporate speaker information such as monologue and discussion to topic detection and summarization process. Bokaei et al. [175] tried to involve speaker information such as monologue and discussion to improve topic detection and summarization process. It proposes a new segmentation algorithm to divide the transcript into functionally consistent parts by considering utterance information. Banerjee et al. [176] proposed an approach to generating abstractive summaries by fusing important content from several utterances. At first, in the text segmentation, meeting transcripts are separated into various topic segments by applying some lexical cohesion based methods and Bayesian unsupervised topic segmentation, where each segment contains utterances on a specific topic. After detecting topic segments, the important set of utterances in a meeting is identified by using multiple features such as basic features, content features, and segment based features in a supervised learning approach. The important utterances are then combined together to generate a one-sentence summary. A just-in-time keyword/topic extraction from meeting transcripts is proposed by Song et al. [177]. The proposed method considers three major factors that make it different from keyword/topic extraction from normal texts. The first factor is the temporal history of the preceding utterances that grants higher importance to recent utterances than older ones, and the second is topic relevance, which focuses only on the preceding utterances relevant to the current utterance. The final factor is the participants. The utterances spoken by the current speaker should be considered more important than those spoken by other partici-

pants. The proposed method considers these factors simultaneously under a graph-based keyword extraction with some graph operations. Experiments on two data sets in English and Korean show that consideration of these factors results in improved performance in keyword extraction from meeting transcripts.

3.2.4.3 Methodology and Dataset

System Architecture: Topic segmentation and topic identification are joint problems in meeting transcripts analysis. Topic segmentation means division of a text or discourse into topically coherent segments. Topic identification means classification of the segments by subject matter (topic). Extracting the topics that best represent the content of the meeting is one of the most important issues concerning meetings. Meeting transcripts derived from spoken multi-party meetings differ from texts written by people because the speech 'understanding' has many difficult challenges such as: high speech recognition error rate, many disfluencies, uncertain boundaries of language units such as sentences, distributed information across utterances from different speakers, low information density, lack of coherence between utterances, and less tidily structured and less restricted in domain textually. Detecting topics and providing summarization is a significant challenge.

This section explains the requirements from a business perspective and how we can build a system to detect topic change in an automated way by using a framework. The proposed architecture has been designed in order to analyze the transcripts (or part of them) of a meeting and to identify the relevant topics discussed. The first language target is English, while French has been considered as an extension. One of the main goals is to detect the topic shift during a conversation. Identifying the topics and the topic shift discussed in a meeting is very important to generate meeting minutes automatically and also to build a mind map of the meeting in real-time. The discussed topics in a meeting can be extracted from meeting transcripts using machine learning methods.

In the literature some studies has been conducted on analysing the meeting transcripts with the aim of detecting topic changes discussed during the conversations and to perform meeting summarization [169], [178], [173], [179]. In these early studies, researchers tend to use different steps before detecting the topics and topic boundaries from a meeting corpus. The system architecture exhibited in Figure 3.19 is the base of the development procedure of the topic detection module in this experiment. The basic structure of the underlying system is exhibited by the framework in Figure 3.20. The usage of the system is organized in two phases. Phase 1 is the collection of recordings of meeting conversations and translation of the meeting audio recordings to texts. Phase 2 comprises the usage of the intelligence and related functions for the detection of the topics discussed in a meeting . Phase 2 is implemented as a REST API that will use four techniques including data pre-processing

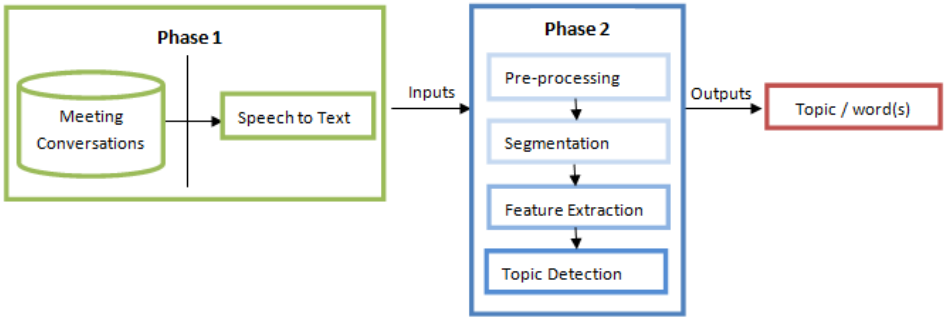


Figure 3.19 – System Architecture of the Meeting Topic Detection.

methods, segmentation techniques, feature extraction methods and topic modelling algorithms.

Meetings will be recorded in real-time in order to fully record the conversations. Then, speech-to-text translation APIs will be used to convert audio into text format and generate the meeting corpus. The output from the Phase 1, the meeting corpus, is the input to the Phase 2 as shown in Figure 3.19. The module will be integrated to work as a real-time module using a REST API. Therefore, it uses the entire meeting corpus generated up to the time of the API request. The main API will collect all the inputs coming from the speech-to-text API at once and then will use all the historical data related to a particular meeting to generate the topics. The algorithm outputs one or more topics based on the meeting corpus generated since the beginning of the discussion.

After the speech is transformed to text in Phase 1, Phase 2 will use that text to detect the topics and topic shifs. As shown in Figure 3.20., there are four different functions (that refer to related best practice techniques) used in the API, they are: data pre-processing methods, segmentation techniques, feature extraction methods and topic modelling algorithms.

Pre-processing is an essential step in NLP and machine learning in order to generate a proper dataset. Pre-processing includes data cleaning, transformation and data reduction methods. In general, meetings corpus includes many discourse markers and therefore, these corpuses are different to the other text corpuses. Hence, we first pre-process the text corpus using following steps in order to perform accurate, efficient and meaningful analysis.
i) Text is converted to all lower case characters.
ii) It is checked against a stop word list (from nltk.corpus import stopwords) - A stop word is a commonly used word (such as the, a, an, in) in sentences, but they are not important while identifying topics in a segment. Therefore, we remove hem using NLTK (Natural Language Toolkit) in python which has inbuilt list of stopwords in different languages. Since we consider both Eglish and French sentences in this experiment, we can directly call stopwords

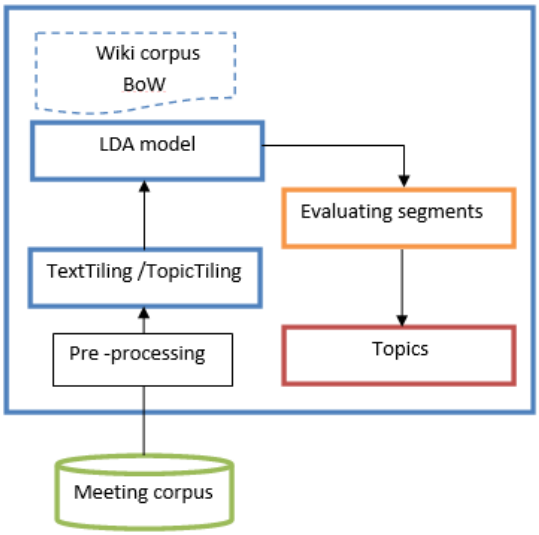


Figure 3.20 – Sofware Architecture of the Meeting Topic Detection.

library in python in our implementation.
iii) Lemmatization is performed (WordNetLemmatizer and TreeTagger) - it is important to identify lemma of a word based in its intended meaning with the knowledge of the context. NLTK library in Python provides WordNetLemmatizer based on Wordnet which is an large, freely and publicly available lexical database for the English language. Therefore, we can directly call NLTK WordNetLemmatizer for detecting lemma of English meeting corpus. As our goal is to implement the CAT module to be portable with French, we used TreeTagger which is a toll that annotates text with part-of-speech and lemma information. TreeTagger supports for about 25 languages including French.)
iv) Text is check against entities like locations, organizations and persons (import polygot) - removing named entities is important from the meeting corpus as they do not provide any valuable features when discussing the topic. Therefore, we used polygot Python library which supports for massive multilingual applications such as language detection (196 Languages), Named Entity Recognition (40 Languages) etc.
v) Removal of the punctuations.

The next step of Phase 2 is to segment the meeting corpus aiming to split into smaller coherent parts. Segmenting a meeting corpus is very challenging due to having incomplete sentences and repetitions in conversations. Moreover, speakers may use pause words, social chatting that are non-verbal, may use visual cues and may have interrupts from the other participants. Then, text needs to be analysed by one 'topic modelling algorithm'.

In general, topic modelling algorithms detect the topic(s) in a collection of text by indexing and characterizing words or collections of words by a probability distribution function. Therefore, topic modelling can be defined as a dimensionality reduction problem and also a clustering problem. For this goal, we use unsupervised machine learning algorithms where we only have input data and no corresponding output variables.

The software architecture of the framework is shown in Figure 3.20. Figure 3.20 indicates the algorithms that have been used and adapted in the topic detection from a meeting corpus. For segmenting the corpus into similar coherent parts, texttiling algorithm and topictilling algorithm are used. Note that, topictilling algorithm is more processing consuming than textilling algorithm. As the main goal of this research is to execute the needed functionalities as a real-time application, the choice is to use textiling algorithm to identify and select the segments. Hence, we use texttiling to generate segments and then identify topics for each segment separately.

Inputs of the Framework: The inputs to the framework shown in 3.20 are as follows.

i) *Data sources*: Framework will use data source directly from meeting transcriptions. Meeting transcription contains strings (integrated in a JSON format) having a set of attributes such as speaker's name, content(Group of words spoken by the speaker), time and MeetingID

ii) *Valid ranges of values*: The main function of the framework only accepts strings and therefore, JSON file contains only string attributes.

iii) *Timing considerations*: Framework will generate the output for each meeting transcript in a timeline, which depends on the way how the attendees are talking. One attendee may use multiple discourse markers leading to a smaller text compared with another attendee who speak constantly. Therefore, main function will trigger when the speech2text returns a string corresponding to a specific time.

Operations done by the Framework: Following describe the set of operations performed within the main function of the framework.

i) *Types of processing required*: In order to generate speech-to-text transcripts we used Google cloud speech-to-text (STT)[6]. However, Google STT cannot transfer speech into multiple text lines or cannot generate multi-line transcripts; instead it returns only one string per translation. Therefore, inputs received to the framework needs pre-processing and segmenting before applying topic detection algorithms to generate a single corpus. When a long string retrieved as the input to the function, it is convenient to understand

[6]https://cloud.google.com/speech-to-text

meeting's progress by segmenting the transcripts leading to retrieve most suitable topic for each segment.

There are three major methodologies that can be used for segmenting text; linear text segmentation, hierarchical text segmentation and function segmentation. In our implementation we used textiling algorithm which will identify unique segments in the corpus, which is a linear text segmentation algorithm. We integrate textual data cleaning and normalization methods prior to applying topic detection algorithms. One of the key algorithms for topic detection is LDA, which is an unsupervised probabilistic machine learning algorithm that is capable of discovering topics based on a training corpus. We adapted LDA in this research to detect the topics for each segment of the meeting transcripts.

ii) *Validity checks*: For each sentence received from the speech-to-text API, framework API evaluates four main parameters (speaker's name, content, time and MeetingID) that are coming as a JSON data. The most important parameters that must be feed in to the framework are 'content' and 'meetingID' and therefore, these two parameters need to be validated before executing next API call for segmenting and topic detection.

Outputs of the Framework: Output generated by the framework will be in a JSON format. REST API is implemented to fetch the results out from the main function of the framework. JSON file includes the attributes such as predicted topic(s), current time and meetingID. By default, predicted topic contains at most 10 words, which is adjustable.

In order to access the output of the framework, another API call has to send a pull request and then, time that a topic returns from the API depends on the querying period of the external REST API. In the remaining period of time, module will be inactive (idle). Error messages and illegal outputs, for instance returning an empty topic, querying without meeting ID etc. are handled by the module itself and will return an JSON error response based on the type of the error.

3.2.4.4 Evaluation and Analysis

As shown in Figure 3.20, at first we need to segment a given meeting transcript in to different meaningful sentences in order to identify the topic shift. For this experiment we first use different textual content to explore how the segmentation takes place.

Figure 3.21 shows one sample dataset we considered as the transcript of a meeting. In Figure 3.21a, the textual content represents under three different topics: explaining about pet, hobby and algorithm. First, our model needs to identify each segments by clustering sentences that are contexualy similar. Hence, we executed the TextTiling algorithm by using the text shown in Figure 3.21a and obtained the result exhibited in Figure 3.21b. Each segment is stated as a block in Figure 3.21b for ease of understanding. We can clearly

```
A pet or companion animal is an animal kept primarily for a person's company, protection, or
entertainment rather than as a working animal, livestock, or laboratory animal.

Popular pets are often noted for their attractive appearances, intelligence, and relatable
personalities. Two of the most popular pets are dogs and cats.

A hobby is a regular activity that is done for enjoyment, typically during one's leisure time.

Hobbies can include collecting themed items and objects, engaging in creative and artistic
pursuits, playing sports, or pursuing other amusements.

A list of hobbies is lengthy and always changing as interests and fashions change.

Engagement in hobbies has increased since the late nineteenth century as workers have more leisure
time and advancing production and technology have provided more support for leisure activities.

In mathematics and computer science, an algorithm is an unambiguous specification of how to solve
a class of problems.

Algorithms can perform calculation, data processing and automated reasoning tasks.

As an effective method, an mathematical algorithm can be expressed within a finite amount of space
and time and in a well-defined formal language for calculating a function
```

(a) Sample trascript explaining 3 different topics: Pet, Hobby and Algorithm.

```
BLOCK NUMBER..... 1
A pet or companion animal is an animal kept primarily for a person's company, protection, or
entertainment rather than as a working animal, livestock, or laboratory animal.

Popular pets are often noted for their attractive appearances, intelligence, and relatable
personalities. Two of the most popular pets are dogs and cats.
BLOCK NUMBER..... 2


A hobby is a regular activity that is done for enjoyment, typically during one's leisure time.

Hobbies can include collecting themed items and objects, engaging in creative and artistic
pursuits, playing sports, or pursuing other amusements.

A list of hobbies is lengthy and always changing as interests and fashions change.

Engagement in hobbies has increased since the late nineteenth century as workers have more leisure
time and advancing production and technology have provided more support for leisure activities.
BLOCK NUMBER..... 3


In mathematics and computer science, an algorithm is an unambiguous specification of how to solve
a class of problems.

Algorithms can perform calculation, data processing and automated reasoning tasks.

As an effective method, an mathematical algorithm can be expressed within a finite amount of space
and time and in a well-defined formal language for calculating a function
```

(b) Result after segmenting the text in Figure 3.21a using TextTiling algorithm .

Figure 3.21 – Text Segmentation.

see from Figure 3.21b that segmentation results provided by the algorithm is very efficient and obtain the result with high accuracy.

In the next phase, we use each segmented text to explore what topics were discussed in a given segment using LDA algorithm. As LDA algorithm is already executed with the segments that can indicate the topic drifts, we can easily identify the topics for each segment separately and also the time that the topic change has been occurred.

The sample dataset shown in Figure 3.21 are randomly chosen text assuming speakers in the meeting are talking about three different topics in different context (pet, hobby and algorithm). However, meeting corpus are somewhat different to this example, but we cannot provide any evidences on how they look-like as those recordings and meeting transcripts are belongs to Orange.

3.2.4.5 Future works

One of the main algorithm used in this work is LDA, which makes use of two assumptions; 1) similar topics will use similar words and 2) documents can be considered as a mixture of multiple words. One major drawback of this model is we must pre-defined the number of topics that need be generated by the model itself. Hence, i order to understand how many topics need to generate for different segments we can use nonparametric approaches such as Dirichlet Process Mixtures. This is one of the main future works that we will adapt. In this technique, instead of pre-defining topic count, model automatically learn it from the text bounded by the unique words in the document that we are considering.

Another future work we are proposing is to use meeting transcripts available in public such as AMI corpus[7]. This corpus consists of multi-model dataset of meeting recordings. The corpus includes recordings from multiple devices in order to track close-talking and far-field. It also used notes and comments on whiteboards and individual pens. This rich dataset will e useful to recognize many properties of the textual content discussed in the meeting in which we do not only need to rely on the spoken text.

3.3 Clickbait Detection with Multimodel Fusion

In this research, we propose a novel approach for clickbait detection which mainly considers many features including content properties and linguistic features of the news post, headline and news article.

[7]http://groups.inf.ed.ac.uk/ami/corpus/

3.3.1 Introduction

In social media, clickbait spread faster than legitimate news and attract much attention from the readers due to its style of presenting the content. These eye catchy contents hard to distinguish as misinformation unless the reader examine the referring article. In social media platform these clickbait are widely used to spread news items. Sometimes, social media post and news headline are different to each other, but use an attractive post with a link to their intended content. We manually observed in some scenarios that news post and the headline share similar content, but the news article itself convey a different message. Therefore, automating clickbait detection is relatively complicated task and a novel research idea. Hence, in order to explore clickbait content in social media we need to focus on several features of different content to achieve significant results.

In the literature (as detailed in Section 2.2) researchers have used several machine learning approaches to detect clickbait in news websites and social media using traditional machine learning approaches. Some research works have focused on detecting clickbait comparing the common words present in the social media post and the news headline [6]. Some other works have used deep learning based techniques [166] and in another study, authors have focused on using random forest classifier to classify clickbait content considering features extracted from image, text and meta feature on Instagram [180].

We propose a clickbait detection model which can utilize multiple features obtained from different textual content (post text, news headline, content of the news article) and integrate them to build a single model for determining the strength of a news content being a clickbait. The proposed clickbait detection model is a fusion model that integrates features such as sentiment score, similarity score and topical similarity of text. We believe that the sentiment score of the social media post and news headline gives a clue on how related they are in terms of the sensational meaning. Also, the topics distribution in the news article and social media post need to be correlated to each other compared to the topical similarity between social media post and news headline. Moreover, similarity score between headline and social media post is also an important factor when determining the clickbait content. The proposed fusion model integrates all these features together to build a single model aiming to classify clickbait content present in the social media. There are several approaches that can be adapted when designing a fusion model such as early fusion, late fusion and hybrid model fusion. In this work, we use late fusion techniques by considering three features mentioned above.

3.3.2 Proposed Fusion Model

In the recent years, multimodel fused based algorithms have gained researchers attention due to the possibility of amalgamating features from different models at different levels such as early fusion and late fusion. Early fusion is mainly dealing with features generated directly from the input data. These input features can be extracted from several unimodels and then concatenate before feeding into the classifier. As all the features generated by unimodels are combined together, it helps to understand the correlation between those features beforehand. One challenge of using early fusion is that different unimodels consists of different frame rates and therefore it is a big challenge to combine those at the same time. Hence, we need to specify a frame rate for achieving this task. In contrast, late fusion uses local class predictors from each unimodel and fuse them into a single vector for ease of classification. Late fusion lacks of understanding the correlation between features, but late fusion is much easier since local decisions made by individual modalities are similar. In this research, we first experiment with late fusion to understand the efficiency of clickbait detection and then apply early fusion to make the prediction more efficient in future research.

Our multimodel clickbait classification approach consists of three main unimodels that are designed for sentiment classification, topic detection and sentence similarity detection. Each of these unimodels are explain in detail below.

3.3.2.1 Unimodel 1 - Sentiment Classification

Sentiment classification is an automated process for identifying opinion of a text in terms of negative, positive or neutral. User feedback helps to understand how they feel about product or service. Understanding emotions from natural language need to apply variety of pre-processing techniques such as tokenisation, lemmatization etc.

In this work, we use our implemented sentiment classification model explained in Section 3.2.3. Main contributions of our sentiment classification model are word-embedding based deep neural network for sentiment classification where we explored Word2Vec or fastText performs better with social media data. In addition to the textutal content, we also try to amalgamate features from emojis as they paly a huge role when deciding a sentiment score. As explained in Section 3.2.3, a set of pre-processing techniques have been adapted and applied to the training dataset such as removing URL, remove multiple characters in a word, remove spaces and dots that are used within words, lemmatization, and we also followed many other simple text processing steps to generate a clean dataset.

Section 3.2.3.3 explains in details about the dataset used to train and test our sentiment classification model. The dataset is collected from Facebook accounts of BBC, CNN and FoxNews. In our experiment, we proposed two different sentiment classification approaches.

We proposed an improved unsupervised sentiment classification model which is implemented considering the lexicons. Since it is important to consider negators, adverbs and model verbs associated with words, we considered them in our implementation and assigned scores for them when defining the final classification. Our unsupervised approach is compared with Vadar sentiment classification approach and our model achieved higher precision for detecting negativity and positivity of a sentence. This model we used as the base model to generate labelled dataset for training an unsupervised model which is based on deep neural networks.

Our supervised sentiment classification model is proposed as a five class sentiment prediction neural network that uses word-embedding to improve the classification accuracy. When defining supervised sentiment classification model, we experimented with several architectural modifications such as training and testing with different layers of CNN, LSTM and combination of both in order to achieve better classification results. The best performed model was using three CNN layer followed by an BiLSTM layer. We observed that Word2vec embeddings performed better than FastText in this analyses and therefore, we used Word2vec in the analyses. This supervised architecture is used as a base model in our clickbait classification task to explore sentiment score of tweets and headlines.

3.3.2.2 Unimodel 2 - Topic Detection

Topics are probabilistic clusters of words in text corpora that are semantically related to each other and they are represented as a set of descriptive and collocated terms. To find these hidden topical patterns of words in a text, topic modelling is used. Topic modelling is a type of statistical model that finds latent variables or hidden structures in the data by applying clustering approach as an unsupervised machine learning technique [181]. The main rule of topic modelling is to discover patterns of word usage and to connect documents that share similar patterns. Since topic model can determine objects as latent topics, i.e., objects that refer to meaning of the collection of documents, it is considered more powerful and applicable than clustering or classification approach [182]. Topic modelling can be applied in individual documents or between some documents that are related.

Clustering approaches for topic modelling provide a good way to classify documents and find topics, but they have a big problem since they just associated each document with one cluster. Therefore, documents composing of multiple topics do not relate to more than one cluster. For tackling with this issue, soft clustering models called Directed Probabilistic Topic Models (DPTMs) are proposed based on generative probabilistic models and hidden topics. All probabilistic models in topic modelling assume that each document consists of a mixture of topics and each topic consists of a set of words, the generative assumption, that are semantically related together through some latent variables. There are several

probabilistic modelling methods such as Latent Semantic Analysis (LSA) [183], Probabilistic latent semantic analysis (PLSA) [184] and Latent Dirichlet Allocation (LDA) [185]. As stated in Section 3.2.4 we use LDA in order to identify the topics precent in the tweet, headline and news article. LDA is one of the most commonly used topic modelling methods which is an unsupervised generative probabilistic method. The basic idea in LDA is that documents are represented as a probabilistic multinomial distribution over latent topics and each latent topic is represented as a probabilistic multinomial distribution over words. It has shown that both the topic distribution in all documents and the word distributions of topics share a common Dirichlet.

3.3.2.3 Unimodel 3 - Similarity Detection

In Section 3.2 we used similarity detection approach proposed during the initial stage of my PhD which is based on the n-grams aiming to identify authorship of written text using SCAP method [106]. Text similarity can be measured in terms of lexical similarity or semantic relationship between two texts. In general, lexical similarity uses string-based, term-based or character-based metrics to evaluate sentences. Whereas, semantic similarity measures are knowledge-based or corpus-based [186]. Some of the widely used lexical similarity techniques includes Longest Common Subsequence (LCS) [187], Jaccard Similarity [188] and N-gram based approaches. Conversely, semantic similarity techniques are Latent Semantic Analysis (LSA) [189] and Explicit Semantic Analysis (ESA) [190]. As the initial stage of our clickbait detection technique, we first experiment with lexical similarity measures and then, in future works, more advanced semantic similarity techniques will be applied.

Lexical similarity measures the similarity degree of two sentences by character matching process. A lexical similarity of 1 means words in both sentences are fully overlapped and 0 means no any common words are in both sentences. String-based algorithms are split into two types: term-based and character-based approaches. In character-based algorithms, counting process is used to count the distance between two sentences. There are many character-based similarity metrics available such as N-gram, Jaro-Winkler, LCS, etc. Term-based similarity measurements calculate number of similar words between two texts such as Euclidean distance, Matching Coefficient, Overlap coefficient and Jaccard coefficient [186]. In this experiment, we use N-gram metrics character-based similarity measure and Jaccard Similarity term-based similarity measure to determine the similarity among news headline and tweet.

3.3.2.4 Framework and Methodology

Multimodel approaches increased model performances in a range of fields, including sarcasm [191], sentiment detection [192] etc. In this research, we apply multimodel fusion

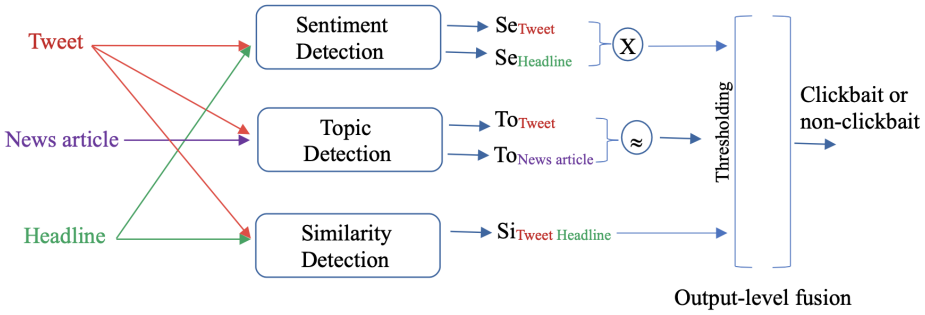


Figure 3.22 – Multimodel clickbait detection architecture.

technique to explore clickbait in social media. As explained in the previous sections, for this analysis, we use three unimodels namely, sentiment detection, similarity detection and topic detection. Figure 3.22 elucidates the architecture of the fusion model with its inputs as tweet, news article and the news headline. We use these inputs to generate different features to evaluate how efficient those features when used for clickbait detection. As shown in Figure 3.22, we use output-level fusion or the late fusion applied through the method of thresholding in such a way that the output from each classifier are pass through a threshold score and applied uniform binning of the raw confidence score. Data binning or discrete binning is used to reduce the minor observation errors in the dataset by grouping a set of continuous values into specified bins.

As shown in Figure 3.22, the first unimodel is sentiment detection, in which we feed the model input as related tweet and headline. The sentiment of the tweet and headline gives a clue on whether they have the similar emotional content. Sometimes, clickbait in social media uses sensational content to redirect users to their web pages, but the actual content is not related to the social media post. Hence, sentiment value of the social media post vs actual headline is an important feature to understand. The sentiment detection algorithm gives the output in 5 classes: very positive, positive, neutral, negative and very negative. We categorize sentiment score as positive if the output received as either positive or very positive and if the model output is negative or very negative, the overall sentiment score is negative. As shown in the following conditional equation, we set the output of the model to 1 if the sentiment value of both tweet and news headline are similar (both sentiment can be positive, negative or neutral) and otherwise, the model output is 0 which indicates that the tweet and news headline are different with respect to their emotional association.

$$f(x_{sentment}) = \begin{cases} 1, & \text{if sentiment of both tweet and news headline are +ve, -ve or neutral} \\ 0, & \text{otherwise} \end{cases}$$

The second unimodel in Figure 3.22 is the similarity score detection which takes the input as tweet and news headline. Legitimate news usually share similar content for its headline and social media post reflecting the meaning of the actual content in the news article, but clickbait are typically use attractive posts that irrelevant to the news article content intending to attract more views. Hence, it is important to identify similarity of the tweet and news headline to observe their relationship in terms of how similar they are. In our experiments, we used both n-gram features and Jaccard coefficient to measure the similarity score and final score is obtained by averaging those two values. If the similarity score is greater than 0.75 we consider those text as similar and assign bin 1, otherwise if they are dissimilar we set value 0 as the output of similarity detection model.

$$f(x_{similarity}) = \begin{cases} 1, & \text{if both tweet and news headline are similar} \\ 0, & \text{otherwise} \end{cases}$$

The next feature is obtained from the topic detection unimodel and inputs to this model are tweet and news article. First, we extract the topics in the news article and then identity whether these topics are reflected in the tweet. As the initial stage, we set 10 number of topics to obtained from the news article. Then, we evaluate whether at least any single topic is presented in the tweet. If there is no any overlapping topics in the news article and the tweet, then the model outputs as 0 and if the topic intersection is positive, then the model output is 1 indicating the tweet reflect some content in the news articles.

$$f(x_{topics}) = \begin{cases} 1, & \text{if topics in tweet and news articles are similar} \\ 0, & \text{otherwise} \end{cases}$$

In the output level fusion, we get scores from each unimodel and then concatenate them to build a vector. This layer finally gives as its output that the probability of a post being clickbait.

3.3.3 Experimental results

3.3.3.1 Dataset

The Webis Clickbait Corpus 2017[8] (Webis-Clickbait-17), the training dataset used in this work, comprises a total of 40,976 Twitter posts in JSON format, obtained from 27 major US

[8]https://webis.de/data/webis-clickbait-17.html

Dataset	#posts	Clickbait:Not
A (Labelled)	2,459	1 : 2.23
B (Labelled)	19,538	1 : 3.10
C (Unlabelled)	18,979	N/A

Table 3.12 – Number of posts and clickbait to non-clickbait ration of each sub-dataset in Webis-Clickbait-17 dataset.

news publishers. JSON files comprised of instances such as post text (tweet), title of target article and description tag of target article. In this work, we used 'postText, targetTitle, targetParagraphs' to decide how clickbaity a given Tweet was. 'postText' represents the textual content of post in twitter without any reference links (we referred this as a social media post), 'targetTitle' indicates the headline or title of the target article (we referred this as the news headline) and 'targetParagraphs' represents text paragraphs in the target article (we referred this as the news article).

Webis Clickbait Corpus dataset has two different labelled datasets (A: 2,495 posts and B: 19,538 posts) and one unlabelled dataset (18,979 posts). Table 3.12 summarizes the exact size of each of these datasets and ratios of clickbait to non-clickbait.

In this dataset Tweets were annotated on a 4-point scale: not click baiting (0.0), slightly click-baiting (0.33), considerably clickbaiting (0.66) and heavily clickbaiting (1.0) by five annotators from Amazon Mechanical Turk. Among all the posts, a total of 9,276 posts were considered clickbait by the majority of annotators based only on the post text. As our aim is to discern clickbait from non-clickbait, we need only the binary classification and hence, we considered clickbait posts as the ones with a score of 0.5 or greater and non-clickbait posts with the score below 0.5. Dataset C, the unlabelled dataset, is only accessible through TIRA evaluation board [9]. We used raw dataset for training without applying any pre-processing techniques. We merged dataset A and B (total of 21,997 labelled data) for training and validating the models. Finally, we compared the performance of our models with the best model proposed at the competition [193]. Dataset consists of two JSON files to represent instance fields and truth scores separately as follows and used some instances from both JSON files for our analysis.

Fields in instances.jsonl:

```
{
    'id': '<instance id >',
    'postTimestamp': '<weekday ><month ><day ><hour >:<minute >:
                        <second ><time_offset ><year >',
```

[9] https://www.tira.io/task/clickbait-detection/dataset/clickbait17-test-170720/

'postText': ['<text of the post with links removed >'],
'postMedia': ['<path to a file in the media archive >'],
'targetTitle': '<title of target article >',
'targetDescription': '<description tag of target article >',
'targetKeywords': '<keywords tag of target article >',
'targetParagraphs': ['<text of the ith paragraph in the target article >'],
'targetCaptions': ['<caption of the ith image in the target article >']
}

Fields in truth.jsonl:

{
'id': '<instance id >',
'truthJudgments': [<number in [0,1] >],
'truthMean': <number in [0,1] >,
'truthMedian': <number in [0,1] >,
'truthMode': <number in [0,1] >,
'truthClass': 'clickbait | no-clickbait'
}

3.3.3.2 Results and observations

We applied the labelled dataset explained in Section 3.3.3.1 for our multimodel fusion architecture. We also compared our model with existing state-of-the-art for the same dataset proposed with other similar approaches. Calculations of these models are calculated on TIRA platform[9] while our algorithms are executed in the two Nvidia 1080 Ti 11GB cards.

Table 3.13 shows the accuracy of the multimodel fusion approach and another model proposed at the Webis clickbait challenge [15]. Our model has achieved 83.97% accuracy in our experimental setup. When compared with the similar type of fusion architecture proposed by Philippe Thomas [194], our model has better accuracy.

Note that this experiment is conducted in early 2018. There are several other proposed approaches at the TIRA platform for the same clickbait competition with better performances than our proposed method. Their primary evaluation metric is the mean square error (MSE) where models are ranked with respect to the mean judgments of the annotators used. In terms of the accuracy (85.6%) and MSE, the best performed model at the competition was introduced by Zhou et al. [66]. Their approach is different to ours and they have used token level self-attentive networks to detect clickbait. However, their approach is based on the labelled dataset and focused only on the postText attribute, and this is one main limitation of their approach even though their results shown to have a slight

Model	**Accuracy(%)**
Multimodel fusion approach	83.97
Philippe Thomas [194]	82.6
Potthast et al. [6] (clickbait17-baseline)	83.24

Table 3.13 – Accuracy of the multimodel fusion approach and other similar approaches proposed at the Webis clickbait challenge [15].

higher accuracy. As per the behaviour of clickbait content, it is much needed to explore different relationship patterns among the image, social media post, news headline and also the content in the news article to explore whether it is a legitimate news or not. Hence, our multimodel fusion approach gives an advantage on this regards where we can integrate several other features extracted from both text and images related to the post.

3.3.4 Future works

The proposed multimodel approach in this work is mainly based on the textual features. In our future works, we aim to incorporate features obtained from images in order to improve the efficiency of the clickbait detection. In addition, early fusion multimodels were shown better results for several applications and therefore, one of the goals in our future works is to apply early fusion techniques to observe the performances of the clickbait detection model by incorporating additional features. Moreover, as per the submitted solutions to the Webis clickbait challenge [15], many proposed models are based on machine learning and neural network approaches. Hence, in our future research we aim to apply recent Transfer Learning models to detect clickbait in social media platforms. Furthermore, we will evaluate and compare the performances of multimodel fusion based techniques and Transfer Learning based detection methods with the focus on clickbait classification.

3.3.5 Conclusion

In this research we proposed a mechanism to detect clickbait using multimodel late fusion approach. The main features considered as the model parameters are similarity among social media post and news headline, sentiment value of social media post vs news headline and topical similarity among social media post and news content. We used the dataset from Webis clickbait challenge [15] that has provided almost 22K labelled dataset including tweet, news headline, news content and many other properties. We applied our multimodel fusion based approach to this dataset and obtained 83.97% accuracy, which is a slight improvement over other similar models proposed at the competition.

3.4 Summary and Discussion

This chapter presented a general overview of the first contribution related to the clickbait detection task. To summarize, it covered how clickbaits can be detected using multimodel fusion technique which mainly considered features extracted from social media post, news headline and news article. The fusion model is based on three other unimodels. The first unimodel is on sentiment classification which is used to identify the sentiment of the social media post vs news headline. The second unimodel is on detecting topics in news article and related social media post. The third unimodel identified the similarity between tweet and the headline. This model is applied on Webis clickbait dataset [15] and achieved 83.97% accuracy.

We also discussed four different experiments that are related to the algorithms used in each unimodel. The first extended experiment is on identifying content originators in social media amalgamating features from user's writing style and online circadian behavior. We proposed a ConOrigina [126] framework and then used this framework to explore news media content interactions in social media. As the second experiment, we observed news content originators, news disseminators and news consumers among news media in Twitter and Facebook. The third experiment is on detecting flaming events (a flood of negative comments targeting a particular social media post) using improved sentiment classification approach. The final extended experiment is on detecting real-time topics discussed in a meeting using topic detection and segmentation techniques. For each contribution of this chapter, a separate related work and future works have been discussed relevant to specific experiment.

In the next chapter, we will discuss one of the main contribution of my thesis, which is about using Transfer Learning to detect clickbait content and methods to optimize those techniques by fine-tuning appropriately.

Chapter 4

Clickbait Detection Using Transfer Learning

Contents

4.1 Introduction

The main focus of this chapter is to propose and adapt recent Transfer Learning (TL) models for clickbait detection task. The general idea of TL is to use the knowledge gained from one task on different but related tasks. The recent attention has focused more on applying TL techniques for NLP applications soon after Google has introduced BERT in 2018 [23] which was initially proposed for masked language modelling (MLM) and next sentence prediction (NSP) tasks. Later, BERT has been improved with the use of large volume of datasets and new design paradigms for achieving better results in different NLP tasks [195]. In this chapter, our focus is on using TL models for clickbait detection and compare performances of these models with respect to traditional machine learning and deep learning techniques. In addition, we also propose several advanced fine-tuning strategies aiming to improve model performances and also to reduce model execution time.

This chapter consists of two main sections. The first section focuses on using three TL models (BERT, XLNet and RoBERTa) for the clickbait detection task and evaluate their performances by integrating several architectural changes to the existing models during fine-tuning. The next section explains different techniques to compress TL models in order to make them faster and smaller than the default architectures without degrading its original performances.

4.2 Clickbait detection using BERT, XLNet and RoBERTa

4.2.1 Introduction

The extensive spread of news in social media is a double-edged sword. On the one hand, social media provides easy access and fast dissemination of news and on the other hand, it allows wide spread of fake news and misinformation within a short period of time. The notoriety of misinformation can be partially attributed to Clickbaits as well. Clickbait is a form of false advertisement designed to attract readers' attention via thumbnail links that lead them to read, view or listen to the content available on the sponsor's web page. Clickbait employs an attractive title or headline that is deceptive, misleading or sensationalized, without indicating the true nature of its 'news' article, aiming to attract many readers and encourage them to click on a link to a web page. A major type of clickbaits are in the form of spam and adverts that are used to redirect users to a web site that sells products or services (often of dubious quality). Another common type of clickbaits are designed to appear as news headlines and redirect readers to their online venues intending to make revenue from page views. In this scenario, a reader can easily be a victim as she assumes the news source to be legitimate, but in reality, this 'news' could be deceptive, sensationalized,

misleading, unverified and irresponsible information [166].

News media in social media are often use clickbaits to generate more value than the actual content by hiding their poor content behind the headline in order to acquire more clicks. Since news media exchange information by acting as both content providers and content consumers [196] [197], misinformation that is deliberately created to mislead requires serious attention due to security and information reliability concerns. Thanks to their very appealing headlines, clickbaits can become popular in online platforms and hence, highly likely to be shared by other users without being checked for legitimacy. Therefore, an automated mechanism to explore and identify the likelihood of a news headline and its content being clickbait would be highly desirable.

Clickbait detection in social media is a challenging task, and only a limited number of research works are available in the literature. Two major competitions were developed to explore various approaches to classify news articles and news items shared in Twitter. The Webis clickbait challenge[1] was designed to develop a classifier that rates how click baiting a Twitter post is. In total, they provide a data corpus having 21,997 items. The other competition was published at the Kaggle in 2019[2] where competitors need to classify articles into news, clickbait and other. Their dataset consists of 24,870 labeled data items for training purposes. We use both these datasets in our analyses to train and evaluate models. Apart from these competitions, a few other recent studies have tackled the clickbait detection task with various state-of-the-art techniques. For example, Potthast et al. [6] conducted an initial study on detecting clickbait titles on Twitter using machine learning methods. They found that top news publishers have employed clickbaits on a regular basis, so that 26% of all the tweets were clickbaits. Some other research works have used deep learning and neural network models [2], [9], [3], [5]. Even with the above mentioned previous approaches, it is very difficult to distinguish between normal headlines and clickbaits. One disadvantage of using deep and machine leaning models in this task is the lack of training data.

One solution to overcome above limitations especially the lack of training data in NLP-related tasks, we can adapt recent Transfer Learning models that can be optimized and improved for clickbait classification through the transfer of knowledge from a related task that has already been learned. In just the past few years, many Transfer Learning models have been introduced by pre-training on different datasets with contrasting architectural modifications. BERT [23], one of the first language representation modeling based on Transfer Learning introduced by Google, achieved state-of-the-art performance on a number of natural language understanding tasks [108]. Subsequently, many other Transfer Learning

[1] https://www.clickbait-challenge.org/
[2] https://www.kaggle.com/c/clickbait-news-detection/overview

models have improved on BERT's performance. In this work, we consider three Transfer Learning models that are widely used in NLP tasks namely, BERT [23], XLNet [24] and RoBERTa [198]. We believe that these models are the representative of most of the other Transfer Learning models in terms of their architectural properties and training datasets. RoBERTa has the same architecture as BERT, but it is pre-trained on a large data corpus, whereas XLNet has a different unique architecture and higher performance than BERT. To the best of authors' knowledge, there have been an insignificant number of attempts to use Transfer Learning models on clickbait detection tasks and there have been no comparative analysis of multiple Transfer Learning models focused on clickbait classification.

Our approach is to apply several fine-tuning strategies on BERT, XLNet and RoBERTa models such as generalization, model compression (pruning) and model expansion to identify the best configuration parameters for clickbait classification. Our main contributions are:
1) adapt Transfer Learning models to improve the detection of news clickbaits in social media,
2) a comprehensive investigation of different fine-tuning strategies on BERT, XLNet and RoBERTa focused on clickbait detection task,
3) a comparison of the performance of our fine-tune models with the best clickbait detection model presented at the Webis clickbait challenge which was based on deep learning approaches, and
4) use an outer-domain dataset (Kaggle dataset) as an application for evaluating generalization capabilities . This last contribution allows us to better understand how the model will behave in a new environment in order to explore generalization capabilities of the proposed models.

4.2.2 Methodology

Typically, clickbait spread on social media are in the form of short messages that refers to certain web content advertisements. Content publishers such as news media revealed clickbait as an effective way of drawing attention to their news websites [6]. After reading such a message, reader get the impression that something is referred to or some emotional reaction is delivered.

An example for a clickbait is: 'Here is What Actually Reduces Gun Violence'. These types of messages easily attract readers' attention and entice them to click on the provided link. Hence, clickbait detection is a challenging task as it is required to observe text syntactic, semantics and associated link references. Therefore, NLP techniques can be adapted to inspect clickbait content in social media.

One of the biggest challenges in NLP related tasks is the lack of training data. In

order to overcome this issue, we can rely on recent Transfer Learning models on NLP tasks such as BERT [1]. BERT is a language representation model introduced by Google in 2018, which is designed to pre-train deep bidirectional representations from unlabeled text using Transformers [22]. Over the last few years, a set of new other algorithms were also proposed by advancing BERT and they outperformed BERT on many NLP benchmark datasets, usually within a large margin [195]. In this study, we will use three popular and representative Transfer Learning models (BERT, XLNet and RoBERTa) for clickbait classification by applying several fine-tuning strategies.

4.2.3 Transfer Learning models for clickbait classification

There are two pre-training objectives that have been successful in Transfer Learning NLP models namely, auto-regressive (AR) and auto-encoding (AE) language models. AR language modelling is not capable of capturing bidirectional context and encode text in unidirectional, either forward or backward, but this has been successful in several downstream tasks such as question answering and sentiment analysis. On the other hand, AE based pre-training models can work with bidirectional context and therefore ease of reconstructing original data from corrupted data. A popular example of such modeling is used in BERT (Google AI) [1]. RoBERTa (Facebook AI) [198] is another model that uses AE language modeling which has the similar architecture as BERT, but pre-trained with a large data corpus. One example for a model which uses AR language modeling is XLNet (Google AI) [24] designed with a unique architecture. One similarity among BERT, XLNet and RoBERTa is that they rely on independent layers stacked on top of each other and use only the Transformer encoder within the model. Due to these architectural similarities and different modeling properties of three models mentioned above, we believe they can be considered as representative models and therefore we will consider them in the clickbait classification task by fine-tuning with appropriate mechanisms. More details about BERT, RoBERTa and XLNet models are given in the following sections.

4.2.3.1 BERT

BERT [1] was the first approach that used deeply bidirectional self-attention mechanism pre-trained on a large data corpus including a Book-Corpus, a dataset consisting of 11,038 unpublished books (plain text corpus) from 16 different genres and 2,500 million words from text passages of English Wikipedia. BERT is designed as a bidirectional contextual model that can consider words' previous and next context and hence, referred to as a contextual model. Contextual models consider the neighboring words in a sentence and therefore has different representations based on the context of the sentence. Whereas, word-embedding

representations like Word2Vec are context-free models in which, one word in two different sentences in different context has the same representation.

There are two main steps in any Transfer Learning model: pre-training and fine-tuning. During pre-training, model-m trains on a dataset A and during fine-tuning, some parameters will be re-used from model-m that was trained on dataset A and trains a model-m on a new dataset B where it transfers knowledge obtained from dataset A to dataset B. In the pre-training phase of BERT, few original tokens in a given sentence replace with mask tokens ([MASK]) and later predict the original sentence using AE language modeling by considering the context of the [MASK] token from both backward and forward directions. In addition, BERT assume that the predicted [MASK] tokens are independent from each other. As a result, in order to obtain a better relationship among all the tokens, it is necessary to have a correlation among unmasked tokens and predicted masked tokens. BERT was trained on many unlabeled data corpora (Table 4.1) considering different scenarios. During fine-tuning BERT, it automatically initialize with the pre-trained parameters.

As explained earlier, BERT uses [MASK] symbol to predict missing tokens. For example, suppose for a text sequence x, BERT constructs a corrupted version as $\hat{x}$ by randomly replacing a set of tokens in x with a symbol [MASK]. If the set of masked tokens are $\bar{x}$, the training objective is to reconstruct $\bar{x}$ from $\hat{x}$ as follows where, $m_t = 1$ indicates that x_t is masked, H is the Transformer that maps a given sentence of length T into hidden vectors.

$$\begin{aligned} max\,(log\,\mathfrak{p}\,(\bar{X}|\hat{X})) &\approx \Sigma_{t=1}^{T}\, m_t\, log\,\mathfrak{p}\,(x_t|\hat{X}) \\ &\approx \Sigma_{t=1}^{T}\, m_t\, log\, \frac{exp(\, H(\hat{X}_t^T)\, e(x_t)\,)}{\Sigma_{\acute{x}}\, exp(\, H(\hat{X}_t^T)\, e(\acute{x})\,)} \end{aligned} \tag{4.1}$$

Two main disadvantages over BERT are;
1) all the masked tokens - $\bar{x}$ and corrupted version - $\hat{x}$ in the joint conditional probability $\mathfrak{p}(\bar{x}|\hat{x})$ are reconstructed separately, and
2) masked tokens are not appeared in the downstream tasks, which creates a pre-train fine-tune discrepancy.

The main advantage of the Auto-encoding language modeling used in BERT is the ability to capture bidirectional context.

There are several approaches to fine-tune BERT. In this study, we modify the architecture of the BERT model during fine-tuning phase by merging additional output layer(s) and also, performing layer pruning to reduce number of layers in the model focusing on clickbait detection task.

4.2.3.2 XLNet

XLNet [24], another Transfer Learning model introduced by Google AI in 2019, is a BERT-like model but generalized with AR method and outperformed BERT on several benchmark datasets [195]. In order to overcome the limitation of AE models, mainly the issue of capturing bidirectional context, XLNet has introduced Permutation Language Modeling (PLM) as explain below. Since XLNet was designed to use permutations of occurrences for a given word, it trains through every possible word in the sequence that take much longer time to converge than BERT.

The main idea of XLNet is to use PLM by adding more features to capture bidirectional contexts. If a sentence has x tokens having length T, then in total $T!$ number of different orders can be constructed to perform AR factorization by considering all positions on both sides of a token. Assume Z_T is the all possible permutation of the sequences having length T.

$$max\ \mathbb{E}_z \backsim z_T[\Sigma_{t=1}^{T}\ log\, \mathfrak{p}\,(x_{z_t}|X_{z<t})] \tag{4.2}$$

where z_t and $z < t$ denotes t-th element and t-1 elements of a permutation Z_T. The XLNet auto-regressive permutation method is shown in equation 4.2 which calculates the probability of token x_{z_t} given preceding tokens $X_{z<t}$ from any order from Z_T. XLNet only permutes the factorization order not the sequence order where it keeps the original sequence order and use Transformers to achieve the positional encoding corresponding to the original sequence. This is a useful property for fine-tuning as it consider only the natural order in a given sequence. The architecture of BERT and XLNet are different in this regard and therefore we use XLNet in our experiments to comparatively analyze different models targeting the clickbait classification.

4.2.3.3 RoBERTa:

RoBERTa (Robustly optimized BERT approach) [198] was developed by improving BERT and therefore, share many similar configurations. RoBERTa outperformed BERT in many benchmark datasets [195]. The main improvements on RoBERTa over BERT are: trained with a large dataset, used dynamic masking patterns and trained on longer sequences. Hence, RoBERTa was improved over BERT by increasing the training data size and hyper-parameters only. RoBERTa used dynamic masking for each training instance in every epoch by duplicating the training dataset 10 times that leads each sequence to be masked in 10 different ways.

A comparison among BERT, XLNet and RoBERTa is presented in Table 4.1. RoBERTa and XLNet used larger mini-batches, learning rates and step sizes for longer training along

		BERT	XLNet	RoBERTa
Number of parameters (Million)	**Base**	110 (12 layers, 12 attention heads, 768 hidden size)	110 (12 layers, 12 attention heads, 768 hidden units)	125 (12 layers, 12 attention heads, 768 hidden units)
	Large	340 (24 layers, 16 attention heads, 1024 hidden size)	340 (24 layer, 16 attention heads, 768 hidden units)	355 (24 layer, 16 attention heads, 768 hidden units)
Performance		Outperforms SoTA in October 2018	2-15% improvement over BERT	2-20% improvement over BERT
Sequence length-SL & batch size-BS during training		SL: 128 for 90% of the steps, 512 for remaining 10% & BS: 256	SL: 512 & BS: 8192	SL: 512 & BS: 256
Data		16GB data Wikipedia+ Books corpus, 3.3 Billion words	Base: 16GB BERT data Large: 16GB BERT+ 16GB Giga5+ 19GB ClueWeb 2012-B+ 110GB Common Crawl , 33 Billion words	16GB BERT + 76GB Common Crawl-News + 38GB Open web text+ 31GB Stories
Method		Bidirectional transformers with Masked Language Model-MLM & Next Sentence Prediction-NSP	Bidirectional transformers with permutation language modelling-PLM	BERT with improved NSP

Table 4.1 – Comparisons of BERT, XLNet and RoBERTa.

with differences in masking procedure [24], [198]. However, pre-training on more data does not guaranteed that the performance of the model will be high and also, very hard to say which model performs better for a specific task that were pre-trained on different datasets. The model performance is mainly based on the number of model parameters, size of the dataset and the amount of computational power that is necessary for training the model and fine-tuning for a new task. Moreover, Talmor et al. [88] found that different models with identical structure and objective functions differ not only quantitatively but also qualitatively. Therefore, it is worth to understand the best model for a clickbait classification by introducing different fine-tuning strategies for each model separately. One main advantage of using RoBERTa and XLNet for the clickbait classification task is that they were pre-trained on Common Crawl news dataset which contains 63 Million English news articles collected between September 2016 and February 2019 and hence, this may helps improve our news clickbait classification models due to these models are pre-trained on news related text content.

4.2.4 Fine-tuning strategies used in this research

As shown in Table 4.1, three models consider in this work are pre-trained on multiple datasets and the ext phase is to apply several fine-tuning strategies focusing on clickbait classification. We shortlisted and proposed eight fine-tuning strategies with the aim of comparing different model performances. These fine-tuning methods can be applied to any Transfer Learning model for any down stream task.

There are several ways to fine-tune Transfer Learning models. We considered three different fine-tuning strategies namely, generalization, compression and expansion. Apart from these fine-tuning strategies, we also use data augmentation methods in order to generate a balanced training dataset. We modified the default architecture of each model to explore the best model parameters for clickbait classification.

4.2.4.1 Generalization:

Generalization is important for Transfer Learning as models train in an unsupervised manner using a large dataset and later, fine-tune as supervised learning approach using a related labeled dataset. Transfer Learning models need to generalize (in-domain generalization or outer-domain generalization) in order to achieve high accuracy, usually by adding an extra task-specific final layer which has to be fine-tuned on a labeled dataset for the task of interest [89]. The accuracy of the model can be improved after generalization by training multiple times on the same supervised data with different random seeds [90]. Generally, distinct random seeds can lead to substantially different results when fine-tuning the model even with the same hyper-parameters. In our experiments, we executed best-performing model multiple times using the same hyper-parameter values, but modifying only the random seed value that control the initialization of the weights of the final classification layer. In our clickbait classification task we will do the generalization and then train multiple times to achieve higher accuracy.

4.2.4.2 Compression:

BERT, XLNet and RoBERTa are exponentially large models since they use huge datasets and millions of parameters during pre-training (Table 4.1). These pre-trained models can be fine-tuned by applying compression techniques to make them smaller and possibly faster. Compression reduces number of parameters in the model in both during-training and post-training. Post-training does not need any training data while during-training rely on training data to decrease number of parameters in the model.

Pruning is one of the compression techniques that modifies the model architecture. There are three different types of pruning strategies namely,
i) head pruning - remove less important heads for a specific task,
ii) weight pruning - remove unnecessary weights in the architecture, and
iii) layer pruning - remove full layer of the transformer and/or dropouts.
In this research, our main focus is on the layer pruning and then assess whether pruning accelerates inference when classifying clickbait.

The base models of BERT, XLNet and RoBERTa consists of 12 transformer layers and 12 attention heads, resulting in a total of 144 unique attention mechanisms. A non-linear feed-forward layer takes the output from each attention head and operate parallel to one another. Therefore, these models can capture a wide range of relationship among the words in a sentence which leads to form a rich representation as it traverse to the deepest layers of the model. Each attention head learns unique parameters and do not share parameters among other attentions.

In general, each attention head is composed of four distinct matrices (Wv, Wo, Wk, Wq) that are generated during training: Wo, Wv - weighted average of output and value vectors and Wq, Wk - query and key vectors that are necessary to compute Wo, Wv having dimension of d vectors. Following equation shows the how attention is used to generate multi-headed attentions.

$$MultiheadAttention(x, q) = \Sigma_{h=1}^{N_h} Attention(W_k^h W_q^h W_v^h W_o^h(x, q)) \tag{4.3}$$

where N_h is the number of independent parametrized attention heads, h indicates an attention head, d_h is the dimension of the head and each head projects down to a different subspace of size d_h where $d_h = \frac{d}{N_h}$, d is the dimension of the input vector, x represents the input, query q to represent a newly computed sequence of representation, $W_k^h W_q^h W_v^h \in \mathbb{R}^{d_h \times d}$ and $W_o^h \in \mathbb{R}^{d \times d_h}$.

To allow all attention heads to interact among them, a non-linear feed-forward layer is used at each transformer layer. Each attention head takes input sequence $x = [x_1, ...x_n]$ corresponding to n tokens and each x_i is transformed into query q_i, key k_i, value v_i and output o_i. Weight pruning and head pruning can be done on these weights which we will explain in Section 4.3.

The base models of BERT, XLNet and RoBERTa consist of 12 Transformer layers with 12 attention heads followed by a feed-forward (FF) sublayer and then, followed by the layer normalization operation. The normalization computes the average and standard deviation of the output activations of a given sublayer and normalizes them. Therefore, the output y_t of a Transformer layer is as follows.

$$y_t = AddNormalization(FF(z_t)) \tag{4.4}$$

$$z_t = AddNormalization(MultiheadAttention(x_t, q)) \tag{4.5}$$

The output of each layer (y_t) is a normalized layer obtained from the output of self-attention layers including residual connections and bias. The entire model stack consists of those 12 layers having a dimension of 768 hidden units. The final transformation is applied on the [CLS] token at the final hidden state which has the size of $d * d$ linear transformation, named as a pooled output.

The default outputs of BERT, RoBERTa, XLNet returns a pooled output of dimension [1, 1, 768] which is the embedding of [CLS] token. We can access linear transformation of each layer that returns from the model as a sequence output of dimension [1, n, 768] where n represents maximum number of tokens. We can also access hidden states of models which is the output of each layer and therefore, each model consists of 12 hidden states. We applied several layer pruning strategies by observing the results from both sequence and hidden outputs of the model.

4.2.4.3 Expansion:

Another experiment we conduct in this research is to analyze behavior of the model and its accuracy after merging extra layers to the output (both pooled and hidden outputs). By default, models use linear layer with the pooled-output as shown by Case 1 in Figure 4.3d. We can expand models by merging additional layers to the latest layer and this will affect model performance when it is used for classification. Hence, we will analyse whether the addition of new layers improves model performances or that leads to degrade the its performances.

4.2.4.4 Data Augmentation

In any classification task, a balanced dataset helps to generate clear decision boundaries with respect to each class and help models to classify data more accurately. Data augmentation techniques can be adapted to make any unbalanced dataset to be a balanced dataset helping to make dataset consistence across different labels.

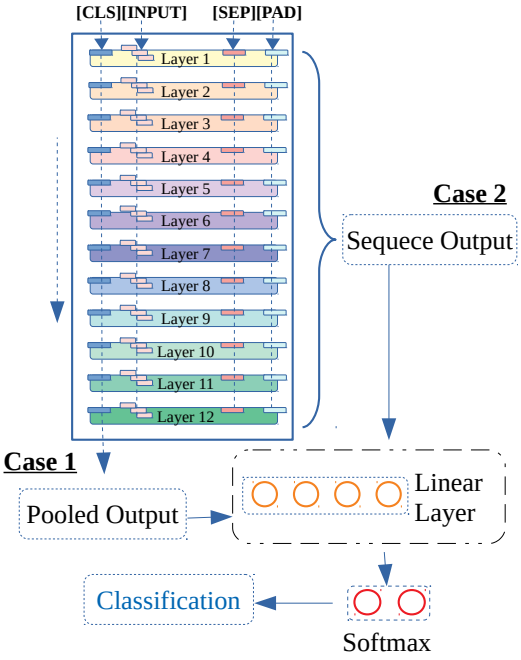
We have research on several data augmentation strategies and SMOTE is one of the popular data augmentation strategy that can easily be adapted to many datasets. Hence, we use SMOTE algorithm [100], can be applied for any dataset without biasing predictions on a specific label. SMOTE over-samples the minority class using k-Nearest Neighbors classifier by selecting samples that are close in the feature space and create synthetic data points. Therefore, in this research we adapt SMOTE to make the dataset balanced across each label and then, evaluate how it affects model performances.

4.2.5 Methodology and experiments

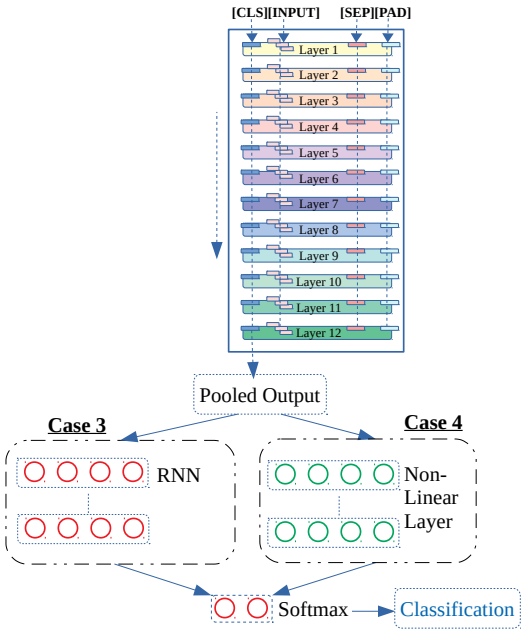
This section discusses about the series of experiments we conduct on clickbait classification by modifying default output parameters of each Transfer Learning model considering the fine-tuning strategies explained in Section 4.2.4. BERT, XLNet and RoBERTa were trained on different datasets including the news media data as detailed in Table 4.1. As our main focus is to investigate clickbait classification approach, generalization is necessary since learning is performed from one domain to another. We use supervised learning techniques to generalize each Transfer Learning model to a clickbait classification task and using two different labeled datasets.

We considered 8 different experiments and entitled them with 8 Cases for ease of referencing them. Figure 4.1 elucidates only 6 configuration changes proposed to modify default BERT, XLNet and RoBERTa models for exploring a better fine-tuning parameters for our generalization task.

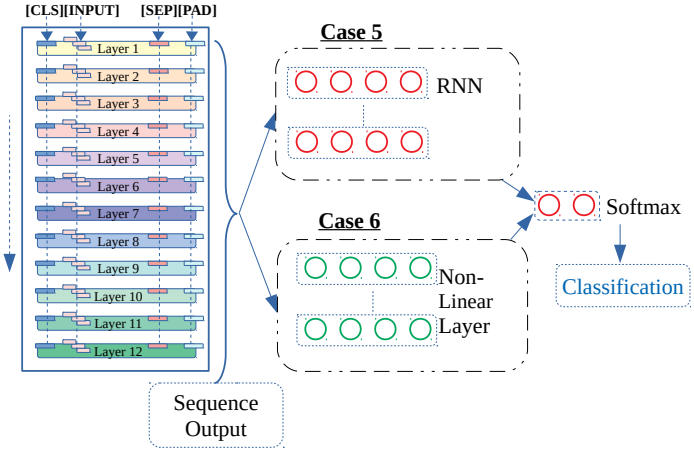
Case 1: The default architecture of BERT, XLNet and RoBERTa is shown with Case



(a) *Case 1*: Pooled output + linear layer. *Case 2*: Sequence output + linear layer



(b) *Case 3*: Pooled output + RNN layer. *Case 4*: Pooled output + non-linear layer



(c) *Case 5*: Sequence output + RNN layer. *Case 6*: Sequence output + non-linear layer

Figure 4.1 – The architectural modifications of the Transfer Learning models.

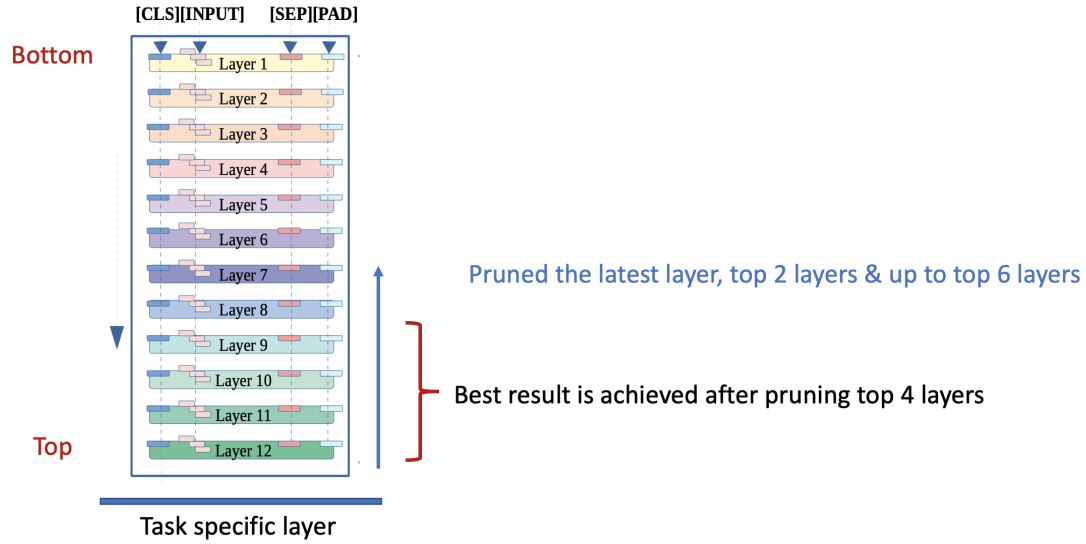


Figure 4.2 – Model architecture for layer pruning that removes one or more top layers from the model.

1 in Figure 4.1a. By default, these models use pooled output, the output of [CLS] token, followed by a linear layer and a softmax layer. The linear layer is a fully connected neural network that projects the vector produced by the [CLS] token in to a large vector called as logits vector. The softmax layer turns logits vector scores into probabilities. We introduced series of experiments to modify the default output of each model as shown in Figure 4.1 from Case 2-6.

Case 2: The only difference between Case 1 and Case 2 is that instead of using pooled output, we concatenated hidden outputs extracted from all 12 layers to make the final vector as shown in Figure 4.1a. With this experiment, we tried to understand whether the output from [CLS] performs better than the output vector generated by considering all hidden outputs.

Case 3: In Case 3, model output integrates with a RNN as elaborated in Figure 4.1b. We used a BiLSTM layer with 2 recurrent layers. In general, BILSTM uses 2 RNNs to have both backward ad forward information where inputs run from past to future ad future to past. BiLSTMs show very good results in the literature as they can understand the context better. When integrating a BiLSTM layer with the Transfer Learning model, it takes the input as the model's output which is the number of encoded tokens return by the model. Hence, two LSTM layers process the input back and forth, and produce a sequence of hidden states which encodes the current token with the previous knowledge and try to integrate

sequential property of the LSTM layer with that. Case 3 explores the performance changes of the model after integrating a BiLSTM layer.

Case 4: In Case 4, instead of RNN layer we added a non-linear layer to the pooled output of the model as shown in Figure 4.1b. One popular non-linear activation function is Rectified Linear Unit - ReLU which has output 0 if the input is less than 0, and raw output otherwise [199]. A major drawback of using ReLU is the dying ReLU problem where some neurons in the model die for all inputs and remain inactive regardless of the any other input we feed to the model. And, if a large number of neurons are inactive, then the performance of the model degrades ad therefore, we ignored using ReLU in our experiments and adapted LeakyReLU, a method which extends the range of ReLU by controlling the angle of the negative slope. Hence, we replaced the default linear layer with a LeakyReLU layer as it allows models for better gradient propagation and efficient computation.

Case 5: Figure 4.1c shows the architectural changes for Case 5 where we consider the sequence output instead of the default pooled output and then, integrate a BiLSTM layer before the classification layer.

Case 6: In Case 6, our aim is to explore model performances when we integrate non-linear layer (LeakyReLU) with the sequence output as indicated in Figure 4.1c.

Case 7: In this research, we pruned model layers in the downstream training to compress the model. Michel et al. [115] showed that up to 40% of attention heads can be pruned from BERT without affecting its test accuracy, and some of these attentions can be entirely span scross a single layer. Hence, it is worth to analyse the performance of the models after compressing on our downstream task after pruning several layer(s). Thus, we try to prune one or more layers, as shown in Figure 4.2, without affecting the output of the model and compared different training loss and performance than the default architecture. This fine-tuning approach is named as Case 7 in our analyses.

Case 8: We observed that the Webis-Clickbait training dataset is imbalanced (as shown in Table 3.12), in which the ratio of clickbait:non-clickbait is almost 1:3. Hence, the validation results and predictions are highly likely to classify as non-clickbait. To overcome this issue, we used data augmentation method called SMOTE that can over-sample the minority class of the dataset, and this experiment is named as Case 8 in our analyses. Case 8 represents the results obtained from models after integrating data augmentation techniques.

4.2.6 Dataset description

In this study, we utilize two different datasets, one for training models and the other dataset is used as testing model that leads us to explore their generalization capabilities.

4.2.6.1 Training Dataset

The Webis Clickbait Corpus 2017[3] (Webis-Clickbait-17), the training dataset used in this work, comprises a total of 40,976 Twitter posts in JSON format, obtained from 27 major US news publishers as explained in Section 3.3.3.1. JSON files comprised of instances such as post text (tweet), title of target article and description tag of target article. In this work, we used only the 'post text' as this was the main attribute considered when deciding how click-baity a given Tweet was. Table 3.12 summarizes the exact size of each of these datasets and ratios of clickbait to non-clickbait.

4.2.6.2 Testing Dataset

We used Kaggle clickbait dataset - 'Train a clickbait detector'[4] as the testing dataset. This clickbait detection task was aimed to classify news articles into three different categories: 'news', 'clickbait' and 'other'. Dataset included a separate training (24,870 entries), testing (5,646 entries) and validation (3,551 entries) sub-datasets. We used their training dataset to evaluate how our models perform in a new environment as our models were trained and fine-tuned using the Webis-Clickbait-17 dataset. In order to prepare this dataset for binary classification we re-labeled 'news' and 'other' categories as 'non-clickbait'. In addition, we pre-processed the testing dataset to eliminate posts that are in different languages and hence, only 18,397 labeled posts are used in the analysis.

4.2.6.3 Fine-tuning BERT, XLNet and RoBERTa

This section includes fine-tuning strategies and experimental results of the clickbait classification downstream task.

Previous solutions to Webis Clickbait challenge were mainly based on the deep neural networks. Creating a very efficient neural network for a classification task can be very expensive as it is required to train using millions of parameters and also need to train the neural network from the scratch at each execution. In addition, it is necessary to have a large corpus of training dataset in order to achieve better performances. Nevertheless, Transfer Learning models execute within much less time than training a neural network for many benchmark tasks [195]. And most importantly, models can use small datasets to train lower layers by fine-tuning on any downstream task. Our modified Transfer Learning algorithms are executed in Google Colab[5]. In order to make fair comparisons with the best-performed model of the Webis Clickbait challenge, proposed by Zhou et al. [14], we executed their model also in the Google Colab.

[3] https://webis.de/data/webis-clickbait-17.html
[4] https://www.kaggle.com/c/dlinnlp-spring-2019-clf
[5] https://colab.research.google.com/notebooks/intro.ipynb

Before fine-tuning BERT, XLNet and RoBERTa, we first tokenized and formatted the input where input sentences are splitted into tokens, prepended with the [CLS] token to the start, appended the [SEP] token to the end, padded to maximum length and finally, created the attention masks. The maximum sequence length is set to 128 on each model, and training-validation split is assigned as 90% for training and 10% for validation.

As clickbait detection is a classification problem, we directly modified and fine-tuned classification classes of each model (*BertForSequenceClassification*, *XLNetForSequenceClassification*, *RobertaForSequenceClassification*) in the 'hugging face'[6] pytorch interfaces.

1) *BertForSequenceClassification* is a Bert Model transformer with a sequence classification/regression head on top (a linear layer on top of the pooled output),

2) *XLNetForSequenceClassification* is a XLNet Model with a sequence classification/regression head on top and

3) *RobertaForSequenceClassification* is a RoBERTa sequence classification model with a linear layer on top of the pooled output.

The outputs of Transfer Learning models are loss, logits (classification) and hidden states (one for the output of the embeddings + one for the output of each layer). After feeding input data to the pre-trained model, additional untrained classification layer will be trained for a target specific task. Hence, one of the main fine-tuning strategies used in this research is to modify the latest layer of the classifier in order to evaluate the performance improvement of the model compared with the default layered architecture as explained in Section 4.2.5.

There are few different pre-trained models exist such as base (12 layers), large (16 layers), uncased (only lower case letters) and cased models. In our experiments we used 'base' models (bert-base-uncased, xlnet-base-cased and roberta-base) in order to make all the architectures comparable. BERT consists of both cased and uncased models, but we observed that the bert-base-uncased model was exhibited higher performance than bert-base-cased and therefore, we used bert-base-uncased model in our experiments. XLNet comprises only a cased model while no any cased or uncased versions introduced for RoBERTa. The training hyper-parameters used in this work includes: learning rate - 2e-5, number of epoch - 6 and batch size - 32 for each execution. In addition to the default fine-tuning strategies, as explained in Section 4.2.2 and as shown in Figure 4.1 we used 8 different architectural modifications including model expansion and pruning, and applying data augmentation techniques.

[6]https://huggingface.co/

Model		MCC	F1 score	Accuracy	Precision	Recall	TP	FP	#Epoch
Case 1	BERT	0.5416	0.6473	0.8368	0.6643	0.6312	63.12	9.92	3
	XLNet	0.5714	0.5649	0.8436	0.4182	0.8704	64.94	9.67	2
	RoBERTa	***0.5727***	***0.6725***	***0.8468***	***0.6824***	***0.6628***	***66.28***	**9.59**	***3***
Case 2	BERT	0.5741	0.6762	0.8448	0.6695	0.6830	68.30	10.49	3
	XLNet	0.5769	0.6852	0.8350	0.6392	0.7383	73.83	13.39	6
	RoBERTa	***0.5947***	***0.6817***	***0.8593***	***0.7358***	***0.6351***	***63.51***	**7.09**	***3***
Case 3	BERT	0.5587	0.6618	0.8418	0.6716	0.6523	65.23	9.92	3
	XLNet	0.5855	0.5697	0.8457	0.4222	0.8755	66.19	9.41	3
	RoBERTa	***0.5882***	***0.5699***	***0.8492***	***0.4219***	***0.8778***	***66.25***	**9.22**	***3***
Case 4	BERT	0.5611	0.6660	0.8402	0.6607	0.6715	67.15	10.73	3
	XLNet	***0.5916***	***0.6938***	***0.8459***	***0.6713***	***0.7178***	***71.78***	**11.29**	***3***
	RoBERTa	0.5885	0.6822	0.8543	0.7071	0.6590	65.90	8.49	4
Case 5	BERT	0.5519	0.6580	0.8380	0.6590	0.6571	65.71	10.58	3
	XLNet	0.5811	0.5738	0.8399	0.4283	0.8692	67.32	10.13	3
	RoBERTa	***0.5843***	***0.6827***	***0.8498***	***0.6843***	***0.6810***	***68.10***	**9.77**	***6***
Case 6	BERT	0.5561	0.6612	0.8395	0.6625	0.6600	66.00	10.46	3
	XLNet	0.5877	0.6889	0.8473	0.6826	0.6953	69.53	10.39	1
	RoBERTa	***0.5982***	***0.6901***	***0.8575***	***0.7130***	***0.6686***	***66.68***	**8.37**	***3***
Case 7	BERT	0.5863	0.6814	0.8530	0.7011	0.6628	66.28	8.79	1
	XLNet	0.5913	0.5704	0.8522	0.4218	0.8809	66.39	8.98	3
	RoBERTa	***0.5922***	***0.6871***	***0.8541***	***0.6994***	***0.6753***	***67.53***	**9.03**	***3***
Case 8	BERT	0.5571	0.6602	0.8416	0.6723	0.6485	64.85	9.83	3
	XLNet	***0.5968***	***0.6915***	***0.8550***	***0.6982***	***0.6849***	***68.49***	**9.21**	***3***
	RoBERTa	0.5574	0.6608	0.8414	0.6706	0.6513	65.13	9.95	5

Table 4.2 – The execution results for eight different fine-tuning cases introduced in Section 4.2.5.

4.2.6.4 Model execution results and observations

In these experiments, we have used the dataset explained in Section 3.3.3.1. The execution results are shown in Table 4.2 using five different metrics: MCC value, accuracy, precision, recall, F1 score, and true positive and false positive values for the validation test. We executed models for 6 epochs and Table 4.2 shows the results for the epoch that received highest number of true positive values. We can observe that many models give their highest true positive values during 3rd epoch. Few observations from the validation results are stated below. As explained in Section 4.2.5, we executed 8 Cases separately and then, we identified which model performed best for each Case based on the performance matrix, mainly considering MCC value and F1 score (the higher the values received for F1 score and MCC, the higher the model performances will be).

Results from model expansion

The result shows that RoBERTa gives highest performance than BERT and XLNet in six cases and XLNet shows better performance in the remaining 2 cases (Case 4 and Case 8). The best performance for RoBERTa is exhibited in Case 2 and Case 6 according to MCC value and accuracy. We noticed that the output of Case 2 and Case 6 were generated by considering the 'hidden outputs' not the default output of the model. RoBERTa shown

		F1 score	Precision	Recall	Accuracy	Execution Environment
Zhou et al.	4-label classification	0.683	0.719	0.650	0.856	TIRA platform
RoBERTa	Case 6	0.690	0.713	0.668	0.858	Google Colab

Table 4.3 – The comparison of the results between the best-performed model at the *Webis Clickbait challenge* and our best-performed Transfer Learning model (The binary classification results in this Table is same as the scores presented for the binary classification in Table 4.2).

higher F1 score in Case 6 than in Case 2. XLNet presented the largest MCC value in Case 4, but its accuracy is lower than many other models considered in our experiments. Apart from that, XLNet exhibited the largest number of false positive values during validation as shown in Table 4.2. As XLNet is based on permutations, the convergence time was much higher than the other models. Hence, we can conclude that RoBERTa performed better in the clickbait classification tasks and its highest performances achieved when we considered the hidden outputs not the default pooled output. We also observed that by expanding RoBERTa adding RNN layers do not significantly improved their performance, especially the false detection, but integrating a non-linear layer to the output has improved model performances.

Next, we executed the best scored model proposed at the Webis clickbait challenge [14] that exhibited the highest accuracy compared to the other proposals in the competition[7]. The model proposed by Zhou et al. [104] was a multi-class classification approach which classifies news content into 4 different classes. They submitted multi-class classification approach to the TIRA platform and achieved 0.856 accuracy as shown in Table 4.3. However, our clickbait classification method is a binary classification approach and the execution platform is Google Colab. Hence, in order to make reasonable comparisons among our proposed models and the Zhou's model, we modified their code to make binary classifications and executed in the Google Colab (the same platform where we executed our Transfer Learning models).

We observed that the performance of the *binary classification* results of the Zhou's model are not satisfactory (only 66.6% accuracy) and shows very low values for the true positive and false positive. We contacted these authors and informed the results we received on their approach for the binary classification, but we have not yet received any feed-back from them. However, the results obtained from our Transfer Learning models are slightly higher than their 4-label classification model executed at the TIRA platform. However, for binary classification Transfer Learning model achieved significant improvement over F1 score compared with the traditional deep learning models. This indicates that the Transfer

[7]https://www.clickbait-challenge.org/

Learning models performed well in the clickbait classification task.

Results from layer pruning

Case 7 in Table 4.2 indicates the performances of the model after applying layer pruning. We conducted a set of experiments by pruning layers in different scenarios.

Top layers in the pre-trained models (refer to Figure 4.2) are usually specialized towards the underlying objective function and therefore, these layers may not important when fine-tuning on a downstream task [200]. We refer top layers as the ones that are closer to the task specific layer. Hence, we experiment by removing the latest layer, top 2 layers, and continued until top 6 layers. After pruning these layer, we merge the task specific layer at the end for making classification. Since, bottom layers of the model are perform local interactions among the words in the input sequence, it is not a good approach to remove those layers. After pruning top layers, we fine-tuned each scenario using the Webis clickbait dataset for four up to 6 epoch.

The best performance is achieved when the models' top 4 layers were pruned and therefore, we ended up with a model which has only 8 layers. We presented this result in Table 4.2. We found that RoBERTa performed well with 8 layers than BERT and XLNet showing better performances.

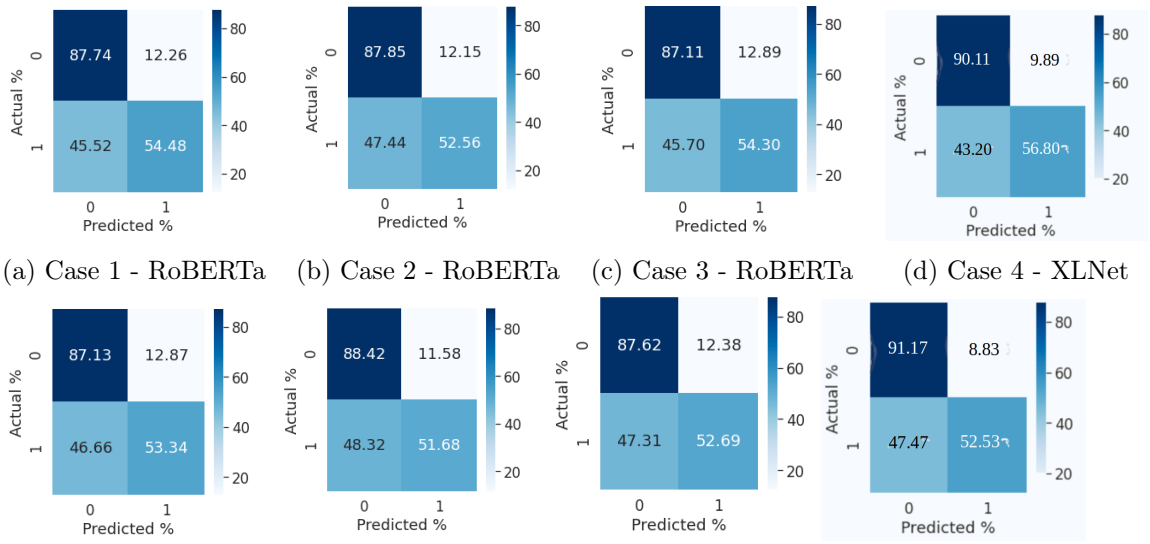
Another major observation that we found from the layer pruning in Case 7 is that, it exhibited a slightly higher accuracy for all BERT, XLNet and RoBERTa compared to the remaining other fine-tuned strategies. Hence, we will expand pruning strategies of these models with novel approaches such as attention pruning and weight pruning in the future studies to improve their performances.

Results from data augmentation

After adding SMOTE for the training dataset, there is no significant improvement on the model performance (shown in Case 8 in Table 4.2) compared to the results for the default model parameters in Case 1. To conclude here, from all the scenarios we considered in our analyses, RoBERTa and XLNet performed better than BERT and in many scenarios RoBERTa performed better than XLNet when fine-tuned by expanding models with a non-linear layer and also generating model output from hidden states.

4.2.6.5 Model generalization performances

Applying Transfer Learning models into a real environment is a challenging since these models were trained with the data extracted from another domain. Even after fine-tuning with a task specific dataset, we cannot ensure they perform well after generalization. Hence, in order to understand how our proposed models are performing in an outer-domain, we



(a) Case 1 - RoBERTa (b) Case 2 - RoBERTa (c) Case 3 - RoBERTa (d) Case 4 - XLNet

(e) Case 5 - RoBERTa (f) Case 6 - RoBERTa (g) Case 7 - RoBERTa (h) Case 8 - XLNet

Figure 4.3 – Evaluation results of the models after generalization in an outer-domain environment (Kaggle clickbait dataset). Label 1 and 0 represents clickbait and non-clickbait, respectively.

used Kaggle dataset published by the task 'Train a clickbait detector', and more details are given in Section 4.2.6.2. This dataset consists of 3,748 clickbait news headlines out of 18,398 samples.

In this experiment, we used the best-performed model from 8 cases and they are highlighted in Table 4.2. Figure 4.3 shows confusion matrix for all Cases after applying Kaggle dataset with these models. We can observe from all confusion matrices that true positive values are always larger than 51% and exhibits at most 13% false positive values and therefore these models performing better at the false detection. However, false negative values are in the rage from 43-47% for all cases. XLNet in Case 4 shows the least false negative value. As a result, we still need to improve model performance with different fine-tuning strategies with a large labeled dataset. To conclude here, at least a half of the clickbait from the Kaggle dataset can be classified with the Transfer Learning models. The best result is achieved from Case 4 which uses XLNet model that built using a bi-directional LSTM layer, and its output is directly taken from the pooled layer.

4.2.7 Concluding remarks

In this work, we have used 8 different Cases using BERT, XLNet and RoBERTa with the aim of exploring a best Transfer Learning model for clickbait classification task. We have used three fine-tuning approaches, namely; model generalization, expansion and compression

(pruning). The analysis has shown that performances of each model with pruning techniques were slightly higher. In the expansion, the best result is achieved when we generated the output from hidden states without directly using pooled output (the default model output). The addition of a new bi-directional LSTM layer do not exhibited any significant improvement over the other configuration changes but, when we changed the non-linear layer to a linear layer models performed better. In addition, we observed that RoBERTa performed better than other two models in many cases. This is obvious that the RoBERTa is pre-trained on a large data corpus than other models and also training data includes news media data as well. RoBERTa has detected the least number of false positive when we fine-tuned it by considering hidden outputs together with non-linear layer at the end. As a result, we can conclude that, model performance can be improved further by experimenting with advanced pruning techniques and considering hidden output parameters of the model. We also observed that, the generalized models are better at the false detection, but still these models can improve their performances by increasing the training dataset.

4.2.8 Possible future experiments

The experiments we conducted in this research were mainly based on the Transfer Learning models. The main fine-tuning and configuration changes we did on BERT, XLNet and RoBERTa are; models are expanded by adding new layer(s) to the existing architecture, considered data augmentation methods and applied layer pruning strategies. In future research works, we will evaluate the performance of models by modifying their architecture by adding changes to the attentions such as add/remove attentions in each layer and keeping necessary weight matrix without dropping them, and pruning based on the weights. Another future research direction is to adapt other features in the dataset such as the headline of the news and keywords, that leads to return correct predictions than considering only the postText. We can consider the syntactic features of the clickbait to understand the correlation between postText, headline and news articles. The trained model then can be used in the real-world dataset to explore clickbait vs non-clickbait content shared by the news media in social media. Apart from that, we will also try to adapt latest Transfer Learning models that are more efficient than the models considered in this work for exploring a better model for clickbait tasks. In addition, since the Webis dataset consists of information about the level of a text being clickbait under four different labels (not click baiting (0.0), slightly click-baiting (0.33), considerably click-baiting (0.66) and heavily click-baiting (1.0)), a novel multi-label classifier model can be proposed to classify under 4 defined classes. Another important research direction is the analysis of fake news in terms of how clickbaity they are and this will help to understand the propagation behaviors of fake news if they are clickbait.

4.3 Clickbait Detection by Compressing BERT

This section explains how we can use advanced fine-tuning strategies such as compression in order to improve clickbait classification with the focus on decreasing inference time and the size of the model while retaining or improving classification accuracy.

4.3.1 Introduction

Natural language processing (NLP), is a form of Artificial Intelligence that helps computer programs to understand, interpret, analyze and manipulate human language, gained much recent attention due to rapid advancements in the processing power, better algorithms and data. The traditional NLP models were mainly based on hand-crafted features in the form of rule-based systems. Two breakthroughs that provided significant enhancements in NLP were Language modelling and Transfer Learning. Language Modelling tasks have to predict the succeeding word given the previous words of a sentence and mainly used for next word predictions tasks and text auto-completion tasks. Transfer Learning is a technique where a model trained on a general task allowed to fine-tune for another specific task to converge faster with relatively low supervised data. Transfer Learning is widely used in applications when lack of large training dataset is available. More details about Transfer Learning is provided in Section 2.5.

Transfer Learning is solely based on Transformers that has become the dominant paradigm in modern machine translation. Multi-head attention is one of the key components of the Transformer model. Different heads in multihead attention potentially focused on different parts of the input and concatenation of their results is used for the predictions in the model. Analysis of multihead attention and identifying the important heads in the model is challenging. In many previous research works on BERT, all attention heads are treated as equally important [201], but some studies have analysed how the attention heads behave within the model and importance of each heads [202]. The main focus of this chapter is to use BERT for detecting clickbait by fine-tuning pre-trained BERT model and also to analyse effect of heads on clickbait detection when compressing the model.

One of the popular compression techniques used in Transformers is pruning, which refers to identifying and removing less important weights or other components in the model to perform better. This research aims to use pruning techniques to remove unnecessary and unimportant attention heads from BERT in order to accelerate inferences while keeping or improving the performance of the model. Section 4.2 detailed on how to adapt BERT, XLNet and RoBERTa Transfer Learning models on clickbait detection and achieved significant improvements on the validation and test datasets, and also considerable performances on the generalization. Therefore, in this work our aim is to introduce several other fine-

tuning strategies that can be applied on the pre-trained BERT model and then, evaluate its performances and inference speed.

4.3.2 Literature Review

Model compression has attracted much attention from researchers and also in the industry as there were many different compression techniques proposed for neural networks (i.e. Convolution Neural Network-CNN) those can directly be adapted to Transformer architectures. Few popular examples of compression techniques include pruning, knowledge distillation, weight quantization and matrix factorization. The general idea of knowledge distillation is, it transfers knowledge from a larger model to a smaller model (i.e. from teacher model to student model) and therefore, requires much more resources to train them. Nevertheless, pruning focuses more on the model itself and reduce number of parameters in it. Pruning a BERT model gained much attention similar to knowledge distillation and therefore, we use pruning in this work which requires less resources compared to other compression techniques [203].

Pruning can be mainly described under three categories [204]: compression, regularization and sparse architecture search. Compression models decreases number of parameters that allows to deployed on low capacity devices and increases inference speed. Pruning a neural Network model also regularizes it where we can consider pruning to be a dropout or heuristic-based L0, L1 regularizes. Pruning can be considers as a sparse architectural search in which pruning makes sparsity explicit.

As described above, most compression techniques for BERT usually adapted knowledge distillation techniques that are based on teacher-student paradigm. With this approach, researchers have introduced several BERT based small size models such as DistilBERT (based on a triple loss combining language modeling, distillation and cosine-distance losses) [205], BERT-PKD (a patient knowledge distillation approach that compresses the original BERT model into a light weight shallow network) [206], Tinybert (adopts a two-stage learning framework: general distillation and task-specific distillation with data augmentation) [207], Distilled BiLSTM (adopts a single layer BiLSTM as the student model) [208], LadaBERT (Lightweight adaptation of BERT through hybrid model compression that combines the advantages of weight pruning, matrix factorization and knowledge distillation.) [209], MobileBERT (a thin version of BERTLarge equipped with bottleneck structures and a carefully designed balance between self-attentions and feed-forward networks) [210].

In general, these techniques require huge resources and take longer time to converge during training those models. Therefore, our aim is to explore more compression techniques that are light weight, and can be applicable for any task specific requirement.

The next widely used BERT compression techniques are usually focused on specific

components of the model instead of compressing the entire model such as element wise pruning and structured pruning [203]. Element wise pruning focused on individual weights. Magnitude weight pruning [204] and re-weighted proximal pruning [211] are examples for Element wise pruning. The structural pruning is more focused on different components of the model such as attention head pruning [83] and encoder unit pruning [212]. There are several other pruning technique have also been used in the literature including neuron pruning, cross pruning, layers pruning, channels pruning etc. The most widely used BERT pruning techniques are magnitude weight pruning and head pruning.

Weight pruning: Magnitude weight pruning is one of the most effective pruning methods. In this technique, when weights represent in some matrix elements are almost zero, then its input can be pruned and ignored without proceeding further. There are three steps that need to be followed when doing magnitude pruning.
1) select which target weights need to be pruned
2) choose a threshold to filter weight magnitude under that threshold and
3) eliminate those weights and train again to achieve if the model loss its previous accuracy.
These pruning strategies can be conducted as a global pruning (considering all network parameters) and matrix local pruning(considering each weight matrix individually).

Head pruning: Head pruning evaluates individual attention heads in the encoder and/or decoder to identify which heads are contributing more on the overall performance of the model and then prune remaining heads to remove without seriously affecting the performance of the model. Voita et al. [202] characterized the roles of the heads in terms of positional, syntactic and attention to rare words or least frequent tokens. They found that, only a small number of heads are important for translation and useful heads have one or more representations in the model.

There are two approaches to perform head pruning and to detect important heads: Layer-wise relevance propagation(LRP) and confidence. Confidence considers the average of maximum attention weights taken over a set of sequences used for evaluation while LPR finds the relative contribution of a head with respect to the other heads in the network. Voita et al. [202] have pruned identified heads in the encoder and decoder on the fine-tuned model. Kovaleva et al. [213] have also explored how different are the self-attention patterns in each head and how much they important for the final task. They calculated similarity between pre-trained and fine-tuned flattened vector of attention weights. Michel et al. [83] tried to explore whether all attentions head are equally important when making predictions. They have suggested that, a large number of heads can be removed when testing the model without degrading the performance of the classification and some layers can be reduced to

a single head. Clark et al. [214] have analyzed different patterns of BERT attention heads such as attending to delimiter tokens, positional offsets, linguistic notions like attending to verbs, nouns, prepositions etc. Work done by Jain et al. [215] explored the relationship between model output and attention weights focusing on several experiments across variety of NLP tasks. Their findings proved that correlation between feature important measures and attention weight is a weak relationship and therefore, attention weights can be changed without altering the model performances.

In this work, our focus is about head pruning on the pre-trained BERT models. It is not possible to conduct research on weight and head pruning during-training as it require exceptionally large computational resources. Hence, these models are costly to train and develop in terms of hardware, computing time, and also these computation are environmentally costly due to the emission of carbon footprint required to process tensor processing applications [216].

4.3.3 Methodology

This section explains step by step process of how BERT head-pruning has been done in this research.

4.3.3.1 Architecture of the Transformer and BERT Model

Transformer proposed by Vaswani et al. [22] has achieved significant improvement on the results for various state-of-the-art NLP tasks. As shown in Figure 2.4, Transformer consists of Encoder and Decoder having different layers such as multihead attention, normalization layer, followed by feed-forward layer. In this section, we briefly describe and provide additional details (the Transformer model introduction is provided in Section 2.5) about the Transformer architecture proposed by Vaswani et al. [22].

Figure 4.4 shows clear representation and relationship among different components of the Transformer. Multi-head attention consists of concatenation of several self-attentions executed in parallel.

$$MultiHead(Q, K, V) = Concat(head_1, head_2, ..., head_h)W^o \tag{4.6}$$

$$where\ head_i = Attention(QW_i^k, KW_i^k, VW_i^k) \tag{4.7}$$

Each head is an attention function of query, key and value vectors. As illustrated in Figure 4.4, there are three types of multi-head attentions available in Transformers: multi-head attention of encoder, masked multi-head attention of decoder and multi-head attention encoder-decoder. Encoder multi-head attention consists of number of self-attention layers and assigned query, key and value vectors generated via input sequence or obtained from

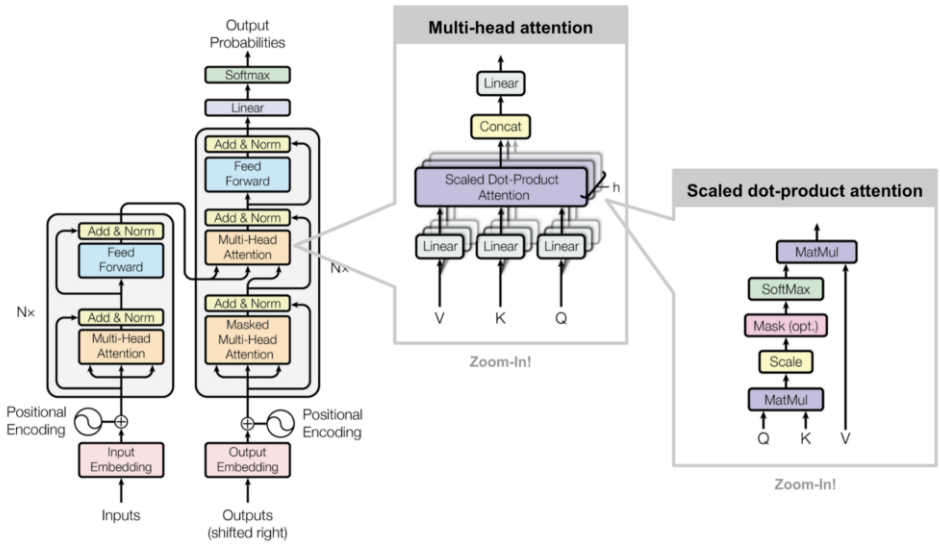


Figure 4.4 – Model architecture of the Transformer and illustration of relationship between different components [22].

the output of the previous encoder. The masked multi-head attention chooses a word and then masked out remaining words after that particular word. This is mainly used to interpret every position in the decoder based on the previous inputs of the sequence. The encoder-decoder multi-head attention retrieve vectors from masked multi-head attention and produce a vector using key and value vectors that are obtained from the top encoder. Self-attention in multi-head attention is the key component of the model architecture and therefore, it is important these attentions to be accurate in order improve efficiencies of tasks. The analysis of multi-head attention and the importance of each neural head for a particular task is challenging.

BERT [23] is a technique introduced by the Google based on Transformers that has achieved significant improvement on the state-of-the-art on many NLP tasks [195]. BERT is pre-trained on 3.3 billion tokens in an unsupervised learning techniques with unlabeled text to perform two main tasks: masked language modeling (MLM) and next sentence prediction (NSP). Models used for MLM predicts a set of words that are masked_out of the input text and NSP predicts whether the second half of a sentence follows the first half. In MLM, BERT masks out at least 15% of the input tokens and 80% these tokens are replaced with [MASK] token, 10% of the token keep as original tokens and remainder replaces with random words. BERT calculate its loss based on how well the model predicts those masked tokens at the end. The 10% of random word replacement and keeping 10% original tokens let the model to learn correct predictions and wrong predictions during training. In the NSP task, BERT uses two consecutive sequences A and B, and then expect

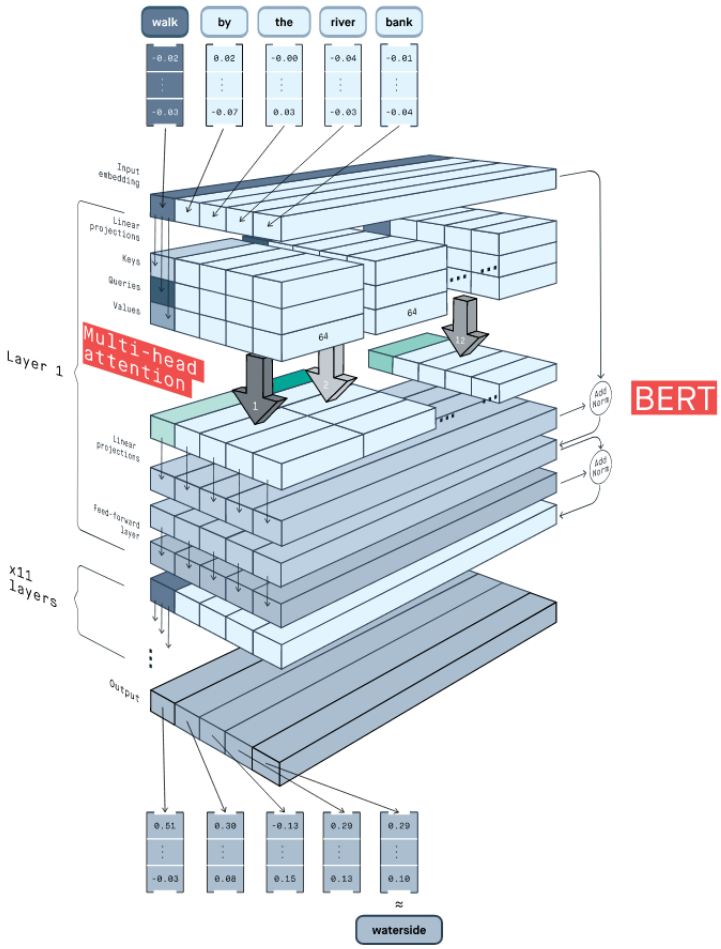


Figure 4.5 – BERT token Encoder stack [217].

the model to predict whether sequence B follows sequence A, or in the next scenario, this A and B selects as two-word sequences for the prediction. Hence, in both tasks BERT uses unlabeled data and allow the model to learn by itself and therefore likely to learn other linguistic patterns such as semantics and co-reference. Training BERT on supervised dataset results performance improvement across various NLP tasks [195].

Figure 4.5 depicts the structure of the BERT model and its components. In this example, the input sequence is 'walk by the river bank'. Each token of the input sequence is replaced by its default embedding vector of 768 length. Figure 4.5 highlighted the 'walk' token to identify how it is proceeded through the model. All the comparisons in the model are done on calculating the scalar product of pair of embedding. Hence, when the two vectors are more correlated to each other or more similar to each other, then the scalar product is higher meaning they have a strong relationship. If they have less similar content, the scalar product

will be lower and therefore they do not related to each other. Model initially calculates the scalar product of every possible pair of the embedding vectors in the input sequence. Then, normalize these vectors, to prevent having large values in the matrix elements. Each scalar product output is divided by the square root of the vector size. Next, each column in the scalar value matrix will pass through a Softmax (as shown in Figure 4.4) to normalize the values. Finally, a new embedding vector is created for each token by the linear combination of the input embedding in proportions given by the Softmax function. This new embedding vectors can therefore called as contextualized vectors since they contain a fraction of every input embedding for a given input sequence. Contextual vector helps to understand when a near by tokens are related to each other as a large fraction of its contextualized embedding will be made of the related embedding. If a token does not related to each other, its contextualized embedding will be nearly identical to the input embedding.

In BERT, input word embedding projects into three linear projections to create key, value and query vectors for each self-attention. Typically, these projections are also mapping the input embedding on to a space of lower dimensions. In the case of BERT key, query and value vectors all have 64 components and each projection can be focusing on different directions of the vector space which represents different semantic aspects of the input sequence. For example, we can imagine that a key is the projection of an embedding on to the directions of prepositions while query is the projection of embedding along the direction of location and the value vector can be generated from another projection. For instance, value vector can be generated based on the embedding of the places. These value vectors are combined to create contextualized embedding. However, in practice, we cannot specifically say which projection is key, value and query vectors are and therefore, model is free to learn any projection focusing on any language specific task. In addition, the same process can be repeated many times with different key, query and value projections for creating multi-head attention. Each head can focus on different projections of the input embedding. For instance, one head can calculate the preposition-location relationship while another head can calculate the subject-verb relationship simply by using the projections to create key, query and value vectors. The output from each head are concatenated and generate a large vector called as head, and BERT uses 12 such heads. Therefore, the final output contains only one 768 contextualized embedding vector per token.

Model also uses positional embedding vectors that contain information about the position in the sequence rather than about the meaning of a token. Hence, attentions calculate relationship among tokens knowing their relative order. Finally, as shown in Figure 4.5, the non-linearity of the Softmax function helps to apply attentions repeatedly and BERT uses 12 layers of such attentions each with its own set of projections. Hence, in total BERT-base uses 144 attention heads and sometimes, all these attention heads may not be useful

for a downstream task in which some attention heads may have a key contribution to the final prediction while other may not be involved and thus, are not important to keep in the computational path. In this work we observe how important each attention head after fie-tuning the model with the clickbait data and then, we disconnect attentions that are not very useful for the final prediction.

4.3.3.2 Methodology and Experiments

Transfer Learning language models such as BERT can be accelerated by applying pruning strategies that allows to generate smaller size yet faster models. The goal of pruning BERT like models is to speed up the inference while retaining the accuracy and also to reduce the size of the model and training time. Some authors have pruned models in the training phase [204] while others mainly applied pruning strategies during fine-tuning. Pruning models during training is more resource consuming task and therefore, our analysis are focused only on pruning BERT in post-training and then, evaluate it with the clickbait classification downstream task.

Dataset: All the experiments in this section uses the Webis Clickbait Corpus 2017 introduced in Section 4.2.6.1. Similar to the previous experiments, we consider only the 'post text' as the main attribute when deciding how clickbaity a given Tweet was. Therefore, as summarized in Table 3.12, we use total of 21,997 labeled data for this classification task. Dataset B in Table 3.12 is used as the training dataset where 90% of this dataset is used for training and the remaining is used for validation, and Dataset A is used as the testing dataset.

Visualizing Attention Heads: BERT-base model consists of 12 layers and each layer has 6 matrices that are prunable (2 matrices as the output of the encoder and 4 matrices as self-attention heads). As explained in section 4.3.3.1, self-attention layers in BERT uses one set of key, query and value vectors at the linear projections for the input sequence, and the next set of key, query, value projection matrices for each attention head. In both pre-trained and fine-tuned BERT models, each layer consists of 12 attention heads that are consistently use specific patterns to encode. In order to illustrate this behavior, as a example, we extracted weight matrices of 3rd layer (we can select any layer) of the first input (i.e. sequence_1) to demonstrate the view of attention heads when processing sequence_1. We generated the Hinton diagram for visualizing weight matrix of 12 attention heads as show in Figure 4.6 where white color represents if the weight matrix index contains a value greater than 0 and grey otherwise.

We observed that, there are few different specific patterns generated by each attention

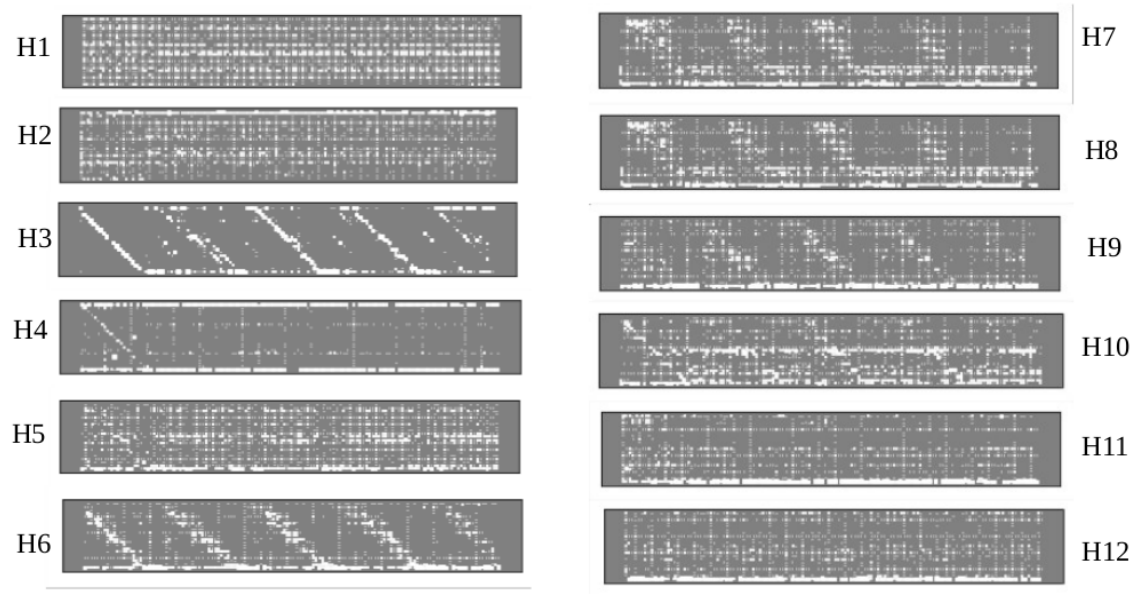


Figure 4.6 – Representation of BERT attentions. Model parameters such as Sequence length and batch size are set to be 64 and 32, respectively. This diagram is generated with weight matrices extracted from BERT Layer 3 for the 1st sequence in the batch generated from Webis Clickbait dataset [15]. The length of the X-axis and Y-axis are 64. A small white color box indicates when particular index in the weight matrix has a value greater than 0.

head during training. We noted that, some of the patterns exhibit vertically that are mainly corresponds to BERT special tokens such as CLS and SEP tokens (e.g, H4, H12 in Figure 4.6). Some patterns can be seen as diagonal which are referring to the previous or following token in the sequence (e.g, H3). Some patterns are exhibited as both vertical and diagonal that are referring to both special tokens and neighboring tokens (e.g, H6). In addition, some other patterns are heterogeneous where these attentions might capture linguistic features rather than only constructing attentions about special tokens and relationship with other tokens in the sentence (e.g, H10).

We can observe from Figure 4.6 that some attention heads may not be very important for the final classification while several other heads impacts a lot. As a result, we try to explore head importance score for each attention head to identify which of them are useful for the final result.

Experimentation setup and Implementation: Our implementation are based on the Huggingface library [218] and executed environment is Google Colab [8]. We use pre-

[8]https://colab.research.google.com

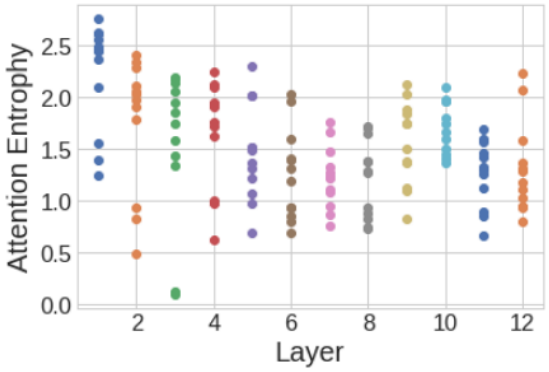


Figure 4.7 – Representation of BERT attentions entropy for each layer.

trained BERT-base-uncased model with 12 layers and 12 attention heads which we fine-tune and evaluate on Webis clickbait dataset [15]. The maximum training sentence length is set to 128 and all models are trained and executed for 4 epoch.

Calculating attention head entropy: In order to observe whether the attention heads focused on a set of tokens or all the tokens of the input, we compute the average entropy of attention heads for each layer. This helps to measure the concentration of attention weights. There are several definitions to the entropy such as Shannon information entropy, Von Neumann entropy etc. In this work, we define the entropy in terms of the probability distribution of attention weights by re-normalizing them. Figure 4.7 shows the average entropy of attention weights of the BERT after fine-tuning using Webis clickbait dataset for 5 epoch.

We can observe that first few layers have very broad attention entropy while layers closer to the output have narrow entropy values and higher the entropy value higher the number of tokens that a particular attention is focused on. If the mean entropy is concentrated, then the attention may focuses on current token position or repeated phrases in the sequence.

Identifying important heads:

In order to prune BERT attention heads, we need to specify additional parameters when initializing pre-trained model to identify what are the important heads that are useful for the final classification. This helps to give priority to each important heads for making the computation more efficient. BERT uses two different masks, the attention_mask and head_mask that we can specify during fine-tuning.

The attention_mask is mainly used to create a mask of 1s (tokens that are not masked) for each token followed by 0s (tokens that are masked) for padding. The head_mask initialized within the forward function of the model which is used to nullify a set of heads

```
Head importance scores
lv, h > 1       2       3       4       5       6       7       8       9       10      11      12
layer 1:        0.00014 0.00037 0.00040 0.00041 0.00077 0.00011 0.00062 0.00009 0.00027 0.00046 0.00036 0.00021
layer 2:        0.00007 0.00019 0.00027 0.00014 0.00030 0.00017 0.00020 0.00017 0.00022 0.00008 0.00013 0.00015
layer 3:        0.00031 0.00019 0.00009 0.00008 0.00014 0.00023 0.00026 0.00009 0.00013 0.00032 0.00022 0.00023
layer 4:        0.00016 0.00010 0.00014 0.00044 0.00021 0.00028 0.00019 0.00013 0.00010 0.00056 0.00021 0.00027
layer 5:        0.00016 0.00009 0.00037 0.00009 0.00024 0.00007 0.00011 0.00011 0.00011 0.00046 0.00022 0.00009
layer 6:        0.00061 0.00021 0.00025 0.00014 0.00040 0.00032 0.00004 0.00018 0.00018 0.00039 0.00029 0.00032
layer 7:        0.00041 0.00048 0.00017 0.00006 0.00055 0.00036 0.00008 0.00045 0.00028 0.00026 0.00015 0.00012
layer 8:        0.00010 0.00035 0.00028 0.00012 0.00009 0.00015 0.00014 0.00008 0.00016 0.00017 0.00028 0.00007
layer 9:        0.00015 0.00014 0.00013 0.00050 0.00011 0.00006 0.00004 0.00067 0.00010 0.00050 0.00038 0.00012
layer 10:       0.00005 0.00030 0.00014 0.00024 0.00035 0.00095 0.00005 0.00025 0.00021 0.00006 0.00039 0.00087
layer 11:       0.00010 0.00019 0.00033 0.00078 0.00011 0.00009 0.00031 0.00039 0.00009 0.00005 0.00008 0.00061
layer 12:       0.00017 0.00009 0.00049 0.00006 0.00023 0.00014 0.00012 0.00009 0.00007 0.00022 0.00022 0.00028
Head ranked by importance scores
lv, h > 1       2       3       4       5       6       7       8       9       10      11      12
layer 1:        90      26      20      19      3       107     5       124     48      15      28      63
layer 2:        133     71      47      88      39      78      68      76      62      127     96      84
layer 3:        37      69      125     126     89      55      49      123     98      35      61      57
layer 4:        82      112     95      17      64      43      72      97      110     8       66      46
layer 5:        80      119     27      122     53      134     106     105     108     14      58      118
layer 6:        7       65      51      87      21      34      142     74      73      24      40      33
layer 7:        18      13      75      136     9       29      128     16      41      50      83      102
layer 8:        111     31      45      100     115     85      94      129     81      77      44      132
layer 9:        86      93      99      10      109     138     143     4       113     11      25      101
layer 10:       141     38      91      54      30      0       140     52      67      137     23      1
layer 11:       114     70      32      2       104     120     36      22      117     139     130     6
layer 12:       79      121     12      135     56      92      103     116     131     60      59      42
```

Figure 4.8 – Head importance score identified through backward pass and heads ranked by the importance score.

of self_attention modules where 1 indicates a head is not masked and 0 indicates a masked head. Hence, in order to prune heads in the self-attentions we first need to identify a set of heads to be removed from the model and then based on those identified heads we need to generate the head_mask.

Equation 4.3 shows how to calculate multihead attention using weight matrices of query, key, value and output vectors. In order to mask heads and identify important attentions, we add a new parameter β to Equation 4.3 [83].

$$MultiheadAttention(x,q) = \Sigma_{h=1}^{N_h} Attention\ \beta\ (W_k^h W_q^h W_v^h W_o^h(x,q)) \tag{4.8}$$

Where β consider as the mask variable (a learnable parameter which is independent from the input sequence) with values in 0,1. We can obtain the same equation as Equation 4.3 when β equal to 1 and when masking attention head h , we set β equals to 0. We apply L0 regularization to β in order to force the model to remove less important heads. This model converged until all less important heads are removed while keeping only very important attentions heads.

First we initialize the head_mask to be null and then build a new head mask from the loss (Cross Entropy loss between the predictions and the passed labels) via performing a backwards pass and updating the weights of the head_mask by calculating the gradient. With this approach, we can identify head important scores and then, based on the important score we can rank heads.

```
Final head mask
lv, h > 1       2       3       4       5       6       7       8       9       10      11      12
layer 1:        0.00000 1.00000 0.00000 1.00000 1.00000 1.00000 0.00000 0.00000 1.00000 1.00000 1.00000 0.00000
layer 2:        1.00000 0.00000 0.00000 0.00000 1.00000 1.00000 1.00000 0.00000 1.00000 1.00000 1.00000 1.00000
layer 3:        1.00000 0.00000 0.00000 1.00000 1.00000 0.00000 1.00000 0.00000 1.00000 1.00000 1.00000 0.00000
layer 4:        0.00000 1.00000 0.00000 1.00000 1.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 1.00000
layer 5:        1.00000 0.00000 1.00000 0.00000 1.00000 1.00000 0.00000 1.00000 0.00000 0.00000 0.00000 0.00000
layer 6:        1.00000 0.00000 1.00000 1.00000 1.00000 1.00000 0.00000 0.00000 0.00000 1.00000 1.00000 0.00000
layer 7:        1.00000 1.00000 0.00000 0.00000 1.00000 1.00000 0.00000 0.00000 1.00000 1.00000 1.00000 0.00000
layer 8:        0.00000 1.00000 1.00000 0.00000 0.00000 0.00000 0.00000 0.00000 1.00000 0.00000 1.00000 0.00000
layer 9:        1.00000 1.00000 0.00000 0.00000 1.00000 0.00000 0.00000 0.00000 1.00000 0.00000 1.00000 1.00000
layer 10:       0.00000 0.00000 1.00000 1.00000 0.00000 1.00000 1.00000 0.00000 1.00000 0.00000 0.00000 1.00000
layer 11:       0.00000 1.00000 1.00000 1.00000 0.00000 0.00000 1.00000 1.00000 0.00000 1.00000 0.00000 1.00000
layer 12:       1.00000 1.00000 1.00000 0.00000 1.00000 0.00000 0.00000 0.00000 0.00000 1.00000 1.00000 0.00000
```

Figure 4.9 – Final head mask after filtering only the important heads.

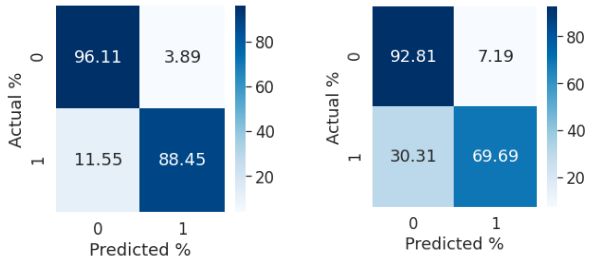
The first head important score matrix calculated with the initial back propagation results are shown in Figure 4.8. In addition, identified heads based on the calculated importance scores are also shown in the same Figure. The results in Figure 4.8 is obtained by masking 10% of the total number of heads. Figure 4.9 shows the final head mask generated after 6 iterations when back propagating weights. We can observe that final head_mask consider only 60 heads indicating 58.3% of the original attention heads considered as unimportant. Hence, we try to modify weight parameters according to the final head_mask and prune the model aiming to reduce number of parameters used within the model.

Pruning fine-tuned BERT: At first, we fine-tuned BERT-base on Webis clickbait dataset for 4 epoch and then used this model as the base for pruning heads. Table 4.4 provides details on the number of parameters and accuracy of the model after pruning and before pruning BERT. The original BERT-base-uncased model has almost 109,482,240 parameters. As shown in Table 4.4, fine-tuned BERT model on clickbait dataset gives 109,483,778 parameters while its pruned BERT model has only 95,707,778 parameters. This means that, after fine-tuning and pruning BERT, model has only 87.42% of the original parameters while neglecting the remaining.

Another interesting observation is that, the validation accuracy of the prediction is almost similar in both before pruning and after pruning, but a slight improvement (only 0.18% increase) on the accuracy after pruning. In addition, as shown in Table 4.4, the model execution time before and after head pruning indicates decrease convergence time after pruning the model. Therefore, after attention head pruning on BERT-base, the validation clickbait dataset used less number of parameters, a slight improvement or similar accuracy and less inference time compared with the same BERT model before compressing. Therefore, we use this compression model in a test dataset to evaluate other performance matrix related to the clickbait classification task.

	Before pruning	After pruning
Number of Parameters	109,483,778	95,707,778
Accuracy (%)	88.39	88.57
Convergence time (hh:mm:ss)	0:02:16	0:01:56

Table 4.4 – Performance measurements of BERT-base (fine-tuned on clickbait dataset) before pruning and after pruning.



(a) Before pruning. (b) After pruning.

Figure 4.10 – Confusion matrix for the Clickbait detection using BERT-base-uncased for before and after attention pruning.
Before pruning - Accuracy: 0.9373, Precision: 0.9108, Recall: 0.8845, F1 score: 0.897470
After pruning- Accuracy: 0.8564, Precision: 0.8131, Recall: 0.6968, F1 score: 0.7505

4.3.3.3 Clickbait detection using pruned BERT-base-uncased model

Next, we use our trained and pruned BERT-base-uncased model to test with a clickbait dataset in order to evaluate how it performs with a new dataset. We used Dataset A in Table 3.12 as the test dataset in this experiment and then executed BERT model separately before compressing and after compressing.

Figure 4.10 depicts the confusion matrix for the test dataset. As shown in Figure 4.10a, before compressing the model, it has achieved 88.45% True Positive values; but after compressing the model True Positive value is only 69.69%. We can also observe an increase of the False Positive after pruning the model. The remaining performance matrices also indicate that, model achieved significant loss in its performance after pruning. Therefore, we can conclude that, even though model compression help to reduce the size of the model and inference time, it does not increase or does not shows a slight fluctuation of its performances especially when we prune the model after fine-tuning. This is an interesting finding to explore in the future work, why the performances degrades considering only the important heads and also the contribution from the unimportant heads for achieving better generalization accuracy.

4.3.4 Concluding remarks

Transfer Learning models have achieved significant improvement over the state-of-the-art for many NLP tasks. These models can be fine-tuned on a downstream task as a supervised learning approach to train the model for that particular task. In this research, our aim is to use Transfer Learning for clickbait detection. We use BERT-base-uncased model and applied set of fine-tuning strategies in which our main focus is on compressing the BERT model to minimize the size of the model and speed up inference. We introduced an attention pruning technique that can compress BERT model by removing unimportant heads after fine-tuning BERT using a clickbait dataset. First, we explored what are the unimportant heads using L0 regularization and then removed them iteratively to generate the final attention mask as a reference for pruning. We observed that, pruning helps to reduce number of parameters in the model, increase validation accuracy and decrease convergence time. However, when evaluating the pruned model with a test dataset or experiment with generalization capabilities, we observed a performance decline when compared with the unpruned model. Hence, pruning a fine-tuned model increased the validation performances, but decreased the test dataset performances in terms of accuracy and F1 score. This is an interesting observation to explore in the future works to understand why model generalization capabilities decreases a lot after pruning. In addition, we will also exploit the behavior of top keywords used in clickbaits within the Transformer model and how much attention they get during training.

Chapter 5

Conclusion and Future Work

Contents

5.1 Conclusion

In this thesis, we propose two different approaches with two contributions to detect clickbaits in social media. Clickbaits can be considered as a form of false advertisement that are used to entice readers to follow a given hyperlink. Many news media sites use clickbaits as a marketing strategy to generate web traffic using sensationalized and misleading content. Therefore, detection of these content is useful in order to prevent social media users. Therefore, our aim is to propose two different approaches to identify clickbait content. In the first contribution, we propose a multimodel fusion approach which combines three features extracted from textual content related to the clickbait. The considered features are extracted from sentiment detection, topic detection and similarity detection algorithms. The next contribution is related to adapting recent Transfer Learning models on the clickbait detection as a downstream task. We present different fine-tuning strategies for BERT, XLNet and RoBERTa in order to fine-tune them to detect clickbait content. In addition, we propose other advanced fine-tune strategies such as model compression, expansion and generalization in order to improve the performance of these model.

5.1.1 Summary and Insights of Contributions

We provide the summary of each contribution, as well as the insights gained from each contribution in this section.

5.1.1.1 Contribution 1

The first contribution aims at proposing a multimodel fusion approach to detect clickbait content. In the proposed approach we used three separate algorithms (similarity, sentiment and topic detection) to extract features from clickbait content. In addition to adapting these algorithms on the clickbait detection, we used them in four other experiments as explained below. The analysis are conducted on the dataset released by the Webis clickbait challenge. Dataset is in the form of JSON files containing attributes such as post text, title, target description, truth class (clickbait or not). The truth class is mainly defied based on the post text. The fusion model uses three features: sentiment value of post text vs title, similarity of title and post text and topic distribution of news article and post text. We generate these features as a one-hot vector and fuse them using late fusion techniques to generate final classification. With this multimodel fusion approach, we achieved 83.97% accuracy for the clickbait detection task.

Content originality detection: Detecting the originator of a content is challenging especially when it is a social media content. We propose an approach to detect social media content originators based on the write print of the user and her online circadian patters.

The proposed framework is known as ConOrigina [126]. Each user has her own style of writing and we use this as a feature to generate a specific knowledge on her writing style. The integration of these two features help to investigate possible authors to a given text based on the historical textual data. More details about this research work can be found from our published paper [126].

News originators and consumers detection in news media community in social media: With growth of social media usage, news media tend to shift their domain from paper work to online media and social media. They advertise their news items in social media, mainly in Facebook and Twitter possibly reaching news items to a large audience at all costs. Therefore, they news media in online venues shown to have a competition among them to publish fresh news and reach to the audience before any another news media. And in some scenarios, news media acts as news aggregators those that collect news items from several other portals. Hence, a set of news media acts as a news originator, another set of news media behave as content providers while the remaining consume news content or replicate the news items shared by another news media. This experiment aimed at exploring these behaviors among news media in Twitter and Facebook using dataset collected from our implemented crawlers. We use ConOrigina [126] to detect text similarity patterns and then provide insights on the behavior of the news media in terms of content originality and published time. More details about this experiment can be found from our published paper [219].

Flaming event detection: Flaming events can be considered as a serious issue in social media in which many users express disagreement, insults and offensive words targeting a particular person or a published post in the form of comments or reviews. Majority of Americans consume news on social media and many of them commonly use Facebook [122]. Therefore, there is a higher chance that the news items shared in Facebook to behave as a flaming event. Sentiment detection can be used to identify the emotional response from the user which then can be used to detect possible flaming events in the news media. We proposed a deep neural network for clickbait classification and also another unsupervised sentiment classification technique to generate the labeled dataset to train the deep model. Using the results obtained from our deep neural network, we identified possible flaming events related to the news items about some political aspects, religious believes and also sensational news items such as murder. Additional details about this experiments can be found from our publication [220].

Detecting the topic change in a real-time meeting: Automatic detection of the topic shift in a real-time meeting is useful mainly to generate the mindmap and also to keep track of the records and inform participants about the current topic they are discussing on. In order to implement an automated mechanism for this, first we need to generate meeting

transcripts by converting audio to text in realtime. Then, we segment these transcripts using an unsupervised manner to separate the spoken coherent sentences into a single cluster. Finally, for each cluster we identified latent topic distribution to generate what topics are discussed in realtime. This experiment was conducted for an industrial project collaborated with my research laboratory[1] and Orange [2].

5.1.1.2 Contribution 2

The second contribution of my thesis on applying Transfer Learning models to detect clickbait content. We conducted two experiments under this contribution as follows.

Usig BERT, XLNet and RoBERTa for clickbait detection: Transfer Learning emerged as a major breakthrough in Natural Language Processing task after introducing BERT in 2018 which achieved significant results for many state-of-the-art NLP tasks. There is a growing research work on applying these models to different downstream tasks. In this experiment, we tried to apply three Transfer Learning models, namely BERT, XLNet and RoBERTa to detect clickbait content. These models have their own architectural changes and therefore we believe these models as the representative models. We proposed fine-tuning strategies such as model compression, model expansion and data augmentation strategies with this experiment. In this experiment, we used layer pruning in order to compress the models. Each model is expanded by using additional layers that are merged to the output layer of the model. Analyses in this experiment are conducted on the Webis clickbait challenge dataset. Apart from that, we used Kaggle clickbait detection dataset as an experimental dataset in order to explore how our fine-tuned models performed in an out-domain datasets. Results have shown that the best performed model is RoBERTa when fine-tuned it by considering hidden outputs and adding a non-linear layer at the final classification.

Clickbait detection with BERT attention pruning: The aim of this experiment is to explore the effects of compressing Transfer Learning models on a specific downstream task. One of the most widely used and popular model compression technique is pruning. In this experiment, we prune unnecessary attention heads in order to reduce the number of parameters and inference speed. We used BERT-base-uncased model and fine-tuned with the Webis clickbait dataset to learn parameters for the downstream task. Then, we mask the unimportant heads in the model and pruned them to keep the useful attention heads within the model. Results indicated that after pruning, BERT model had less number of parameters, slight improvement(almost the same accuracy) to the validation accuracy and less convergence time than before pruning the model. Therefore, we applied our pruned

[1] https://dice.wp.telecom-sudparis.eu/
[2] https://www.orange.fr

model on a test dataset to evaluate its performances. We observed a drastic decline over the performances of the model in the pruned BERT than un-pruned BERT indicating that pruning Transfer Learning model after fine-tuning does not give an additional advantage over the performances in an outer domain dataset.

5.2 Future Work

In the previous chapters, we provided possible future works on different experiments under respective sections. The recent trend on NLP is to apply and adapt Transfer Learning targeting a particular downstream task. Therefore, in this section we propose future works targeting Transfer Learning models. The main focus of our research is to adapt Transfer Learning for clickbait classification, but proposed fine-tuned strategies can be utilized for any other task.

BERT compression is still in the early stage. There are still unrecognized and interesting patters in the BERT model and it is interesting to explore how to use such patterns for better compression. Another interesting research question is to explore the behavior of the model when compressing specific encoder units (such as lower encoder, upper encoder etc.) and varied number of attention heads with different hidden sizes across encoder units. This helps to understand a detailed overview of the classification tasks. In addition, there exists many other compression techniques that were used for neural networks (CNN and BiLSTM) and we can adapt these strategies to BERT. Many compression techniques focus only on specific component of the model to prune, but we can combine several techniques together to achieve better overall compression results such as integrating quantization with weight pruning and attention head pruning. In addition to applying compression techniques only for BERT, it is better to explore other recent Transfer Learning models for clickbait detection.

References

[1] M. Wall, "Citizen journalism: A retrospective on what we know, an agenda for what we don't," *Digital Journalism*, vol. 3, no. 6, pp. 797–813, 2015.

[2] V. Kumar, D. Khattar, S. Gairola, Y. Kumar Lal, and V. Varma, "Identifying clickbait: A multi-strategy approach using neural networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 1225–1228.

[3] M. Dong, L. Yao, X. Wang, B. Benatallah, and C. Huang, "Similarity-aware deep attentive model for clickbait detection," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2019, pp. 56–69.

[4] A. Anand, T. Chakraborty, and N. Park, "We used neural networks to detect clickbaits: You won't believe what happened next!" in *European Conference on Information Retrieval*. Springer, 2017, pp. 541–547.

[5] A. Omidvar, H. Jiang, and A. An, "Using neural network for identifying clickbaits in online news media," in *Annual International Symposium on Information Management and Big Data*. Springer, 2018, pp. 220–232.

[6] M. Potthast, S. Köpsel, B. Stein, and M. Hagen, "Clickbait detection," in *European Conference on Information Retrieval*. Springer, 2016, pp. 810–817.

[7] R. S. O. T. A. Rappoport and M. Koppel, "Authorship attribution of micro-messages," 2013.

[8] M. M. U. Rony, N. Hassan, and M. Yousuf, "Diving deep into clickbaits: Who use them to what extents in which topics with what effects?" in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, 2017, pp. 232–239.

[9] H.-T. Zheng, J.-Y. Chen, X. Yao, A. K. Sangaiah, Y. Jiang, and C.-Z. Zhao, "Clickbait convolutional neural network," *Symmetry*, vol. 10, no. 5, p. 138, 2018.

[10] Y. Ha, J. Kim, D. Won, M. Cha, and J. Joo, "Characterizing clickbaits on instagram," in *Twelfth International AAAI Conference on Web and Social Media*, 2018.

[11] M. Glenski, E. Ayton, D. Arendt, and S. Volkova, "Fishing for clickbaits in social images and texts with linguistically-infused neural network models," *arXiv preprint arXiv:1710.06390*, 2017.

[12] L. Shang, D. Y. Zhang, M. Wang, S. Lai, and D. Wang, "Towards reliable online clickbait video detection: A content-agnostic approach," *Knowledge-Based Systems*, vol. 182, p. 104851, 2019.

[13] S. Zannettou, S. Chatzis, K. Papadamou, and M. Sirivianos, "The good, the bad and the bait: Detecting and characterizing clickbait on youtube," in *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2018, pp. 63–69.

[14] Y. Zhou, "Clickbait detection in tweets using self-attentive network," *arXiv preprint arXiv:1710.05364*, 2017.

[15] B.-U. Weimar, "Webis clickbait challenge," https://www.clickbait-challenge.org/, 2020.

[16] M. Potthast, T. Gollub, K. Komlossy, S. Schuster, M. Wiegmann, E. P. G. Fernandez, M. Hagen, and B. Stein, "Crowdsourcing a large corpus of clickbait on twitter," in *Proceedings of the 27th international conference on computational linguistics*, 2018, pp. 1498–1507.

[17] M. Potthast, T. Gollub, M. Hagen, and B. Stein, "The clickbait challenge 2017: towards a regression model for clickbait strength," *arXiv preprint arXiv:1812.10847*, 2018.

[18] iPavlov research group, "Clickbait news detection," https://www.kaggle.com/c/clickbait-news-detection, 2020.

[19] A. Pujahari and D. S. Sisodia, "Clickbait detection using multiple categorisation techniques," *Journal of Information Science*, p. 0165551519871822, 2019.

[20] H. Jwa, D. Oh, K. Park, J. M. Kang, and H. Lim, "exbake: Automatic fake news detection model based on bidirectional encoder representations from transformers (bert)," *Applied Sciences*, vol. 9, no. 19, p. 4062, 2019.

[21] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[24] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Advances in neural information processing systems*, 2019, pp. 5754–5764.

[25] X. Liu, P. He, W. Chen, and J. Gao, "Multi-task deep neural networks for natural language understanding," *arXiv preprint arXiv:1901.11504*, 2019.

[26] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune bert for text classification?" in *China National Conference on Chinese Computational Linguistics*. Springer, 2019, pp. 194–206.

[27] C. Zhang and M. Abdul-Mageed, "Multi-task bidirectional transformer representations for irony detection," *arXiv preprint arXiv:1909.03526*, 2019.

[28] S. Tang, Q. Liu, and W.-a. Tan, "Intention classification based on transfer learning: A case study on insurance data," in *International Conference on Human Centered Computing*. Springer, 2019, pp. 363–370.

[29] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, "Defending against neural fake news," in *Advances in Neural Information Processing Systems*, 2019, pp. 9051–9062.

[30] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[31] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[32] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 160–167.

[33] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[35] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.

[36] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," *arXiv preprint arXiv:1508.06615*, 2015.

[37] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu, "Charagram: Embedding words and sentences via character n-grams," *arXiv preprint arXiv:1607.02789*, 2016.

[38] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[39] ——, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[40] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[42] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[43] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[44] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[45] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[46] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[47] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," 2011.

[48] N. Srivastava and R. R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," in *Advances in neural information processing systems*, 2012, pp. 2222–2230.

[49] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.

[50] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 2, pp. 423–443, 2018.

[51] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[52] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[53] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[54] S. K. D'mello and J. Kory, "A review and meta-analysis of multimodal affect detection systems," *ACM Computing Surveys (CSUR)*, vol. 47, no. 3, pp. 1–36, 2015.

[55] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[56] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," 2011.

[57] D. Wang, P. Cui, M. Ou, and W. Zhu, "Deep multimodal hashing with orthogonal regularization," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[58] C. Silberer and M. Lapata, "Learning grounded meaning representations with autoencoders," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014, pp. 721–732.

[59] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko, "Translating videos to natural language using deep recurrent neural networks," *arXiv preprint arXiv:1412.4729*, 2014.

[60] S. Chen and Q. Jin, "Multi-modal dimensional emotion recognition using recurrent neural networks," in *Proceedings of the 5th International Workshop on Audio/Visual Emotion Challenge*, 2015, pp. 49–56.

[61] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, "Devise: A deep visual-semantic embedding model," in *Advances in neural information processing systems*, 2013, pp. 2121–2129.

[62] R. Kiros, R. Salakhutdinov, and R. S. Zemel, "Unifying visual-semantic embeddings with multimodal neural language models," *arXiv preprint arXiv:1411.2539*, 2014.

[63] S. K. D'mello and J. Kory, "A review and meta-analysis of multimodal affect detection systems," *ACM Computing Surveys (CSUR)*, vol. 47, no. 3, pp. 1–36, 2015.

[64] P. K. Atrey, M. A. Hossain, A. El Saddik, and M. S. Kankanhalli, "Multimodal fusion for multimedia analysis: a survey," *Multimedia systems*, vol. 16, no. 6, pp. 345–379, 2010.

[65] Z.-Z. Lan, L. Bao, S.-I. Yu, W. Liu, and A. G. Hauptmann, "Multimedia classification and event detection using double fusion," *Multimedia tools and applications*, vol. 71, no. 1, pp. 333–347, 2014.

[66] G. A. Ramirez, T. Baltrušaitis, and L.-P. Morency, "Modeling latent discriminative dynamic of multi-dimensional affective signals," in *International Conference on Affective Computing and Intelligent Interaction*. Springer, 2011, pp. 396–406.

[67] E. Shutova, D. Kiela, and J. Maillard, "Black holes and white rabbits: Metaphor identification with visual features," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 160–170.

[68] E. Morvant, A. Habrard, and S. Ayache, "Majority vote of diverse classifiers for late fusion," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2014, pp. 153–162.

[69] N. Neverova, C. Wolf, G. Taylor, and F. Nebout, "Moddrop: adaptive multi-modal gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 8, pp. 1692–1706, 2015.

[70] Q. Jin and J. Liang, "Video description generation using audio and visual cues," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, 2016, pp. 239–242.

[71] M. Malinowski, M. Rohrbach, and M. Fritz, "Ask your neurons: A neural-based approach to answering questions about images," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1–9.

[72] M. Wöllmer, A. Metallinou, F. Eyben, B. Schuller, and S. Narayanan, "Context-sensitive multimodal emotion recognition from speech and facial expression using bidirectional lstm modeling," in *Proc. INTERSPEECH 2010, Makuhari, Japan*, 2010, pp. 2362–2365.

[73] M. A. Nicolaou, H. Gunes, and M. Pantic, "Continuous prediction of spontaneous affect from multiple cues and modalities in valence-arousal space," *IEEE Transactions on Affective Computing*, vol. 2, no. 2, pp. 92–105, 2011.

[74] Miro, "Nlp model selection," https://miro.com/app/board/o9J_ksdMmCY=/, 2020.

[75] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in nlp," *arXiv preprint arXiv:1906.02243*, 2019.

[76] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[77] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[78] H. Tsai, J. Riesa, M. Johnson, N. Arivazhagan, X. Li, and A. Archer, "Small and practical bert models for sequence labeling," *arXiv preprint arXiv:1909.00100*, 2019.

[79] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: The impact of student initialization on knowledge distillation," *arXiv preprint arXiv:1908.08962*, 2019.

[80] R. Tang, Y. Lu, L. Liu, L. Mou, O. Vechtomova, and J. Lin, "Distilling task-specific knowledge from bert into simple neural networks," *arXiv preprint arXiv:1903.12136*, 2019.

[81] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "Tinybert: Distilling bert for natural language understanding," *arXiv preprint arXiv:1909.10351*, 2019.

[82] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," *arXiv preprint arXiv:1905.09418*, 2019.

[83] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one?" in *Advances in Neural Information Processing Systems*, 2019, pp. 14 014–14 024.

[84] Z. Wang, J. Wohlwend, and T. Lei, "Structured pruning of large language models," *arXiv preprint arXiv:1910.04732*, 2019.

[85] A. Fan, E. Grave, and A. Joulin, "Reducing transformer depth on demand with structured dropout," *arXiv preprint arXiv:1909.11556*, 2019.

[86] X. Team, "A fair comparison study of xlnet and bert with large models," https://medium.com/@xlnet.team/a-fair-comparison-study-of-xlnet-and-bert-with-large-models-5a4257f59dc0, 2019.

[87] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[88] A. Talmor, Y. Elazar, Y. Goldberg, and J. Berant, "olmpics–on what language model pre-training captures," *arXiv preprint arXiv:1912.13283*, 2019.

[89] D. Yogatama, C. d. M. d'Autume, J. Connor, T. Kocisky, M. Chrzanowski, L. Kong, A. Lazaridou, W. Ling, L. Yu, C. Dyer, *et al.*, "Learning and evaluating general linguistic intelligence," *arXiv preprint arXiv:1901.11373*, 2019.

[90] R. T. McCoy, J. Min, and T. Linzen, "Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance," *arXiv preprint arXiv:1911.02969*, 2019.

[91] B. M. Lake and M. Baroni, "Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks," *arXiv preprint arXiv:1711.00350*, 2017.

[92] D. Hupkes, V. Dankers, M. Mul, and E. Bruni, "Compositionality decomposed: How do neural networks generalise?" *Journal of Artificial Intelligence Research*, vol. 67, pp. 757–795, 2020.

[93] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[94] Y. L. N. G. M. G. A. M. O. L. V. S. Lewis, Mike and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *Facebook AI*, 2019.

[95] N. S. A. R. K. L. S. N. M. M. Y. Z. W. L. Raffel, Colin and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Google*, 2019.

[96] J. Phang, T. Févry, and S. R. Bowman, "Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks," *arXiv preprint arXiv:1811.01088*, 2018.

[97] C. Lee, K. Cho, and W. Kang, "Mixout: Effective regularization to finetune large-scale pretrained language models," *arXiv preprint arXiv:1909.11299*, 2019.

[98] H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and T. Zhao, "Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization," *arXiv preprint arXiv:1911.03437*, 2019.

[99] X. Liu, P. He, W. Chen, and J. Gao, "Improving multi-task deep neural networks via knowledge distillation for natural language understanding," *arXiv preprint arXiv:1904.09482*, 2019.

[100] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[101] R. Motamedi, R. Gonzalez, R. Farahbakhsh, A. Cuevas, R. Cuevas, and R. Rejaie, "Characterizing group-level user behavior in major online social networks," Technical Report available at: http://mirage. cs. uoregon. edu/pub/CIS-TR-2013-09. pdf, Tech. Rep., 2014.

[102] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," National Bureau of Economic Research, Tech. Rep., 2017.

[103] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *Communications of the ACM*, vol. 42, no. 2, pp. 39–41, 1999.

[104] P. Tong, J. Yao, L. Wang, and S. Yang, "Comprehensive graph and content feature based user profiling," in *Australasian Database Conference*. Springer, 2016, pp. 31–42.

[105] W.-Y. Lim, J. Goh, and V. L. Thing, "Content-centric age and gender profiling," *Proceedings of the Notebook for PAN at CLEF*, 2013.

[106] G. Frantzeskou, E. Stamatatos, S. Gritzalis, C. E. Chaski, and B. S. Howald, "Identifying authorship by byte-level n-grams: The source code author profile (scap) method," *International Journal of Digital Evidence*, vol. 6, no. 1, pp. 1–18, 2007.

[107] E. Stamatatos, "A survey of modern authorship attribution methods," *Journal of the American Society for information Science and Technology*, vol. 60, no. 3, pp. 538–556, 2009.

[108] S. Afroz, A. C. Islam, A. Stolerman, R. Greenstadt, and D. McCoy, "Doppelgänger finder: Taking stylometry to the underground," in *2014 IEEE Symposium on Security and Privacy*. IEEE, 2014, pp. 212–226.

[109] M. Almishari, D. Kaafar, E. Oguz, and G. Tsudik, "Stylometric linkability of tweets," in *Proceedings of the 13th Workshop on Privacy in the Electronic Society*. ACM, 2014, pp. 205–208.

[110] R. Overdorf and R. Greenstadt, "Blogs, twitter feeds, and reddit comments: Cross-domain authorship attribution," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 3, pp. 155–171, 2016.

[111] S. Afroz, M. Brennan, and R. Greenstadt, "Detecting hoaxes, frauds, and deception in writing style online," in *2012 IEEE Symposium on Security and Privacy*. IEEE, 2012, pp. 461–475.

[112] R. Layton, P. Watters, and R. Dazeley, "Authorship attribution for twitter in 140 characters or less," in *Cybercrime and Trustworthy Computing Workshop (CTC), 2010 Second*. IEEE, 2010, pp. 1–8.

[113] H. Azarbonyad, M. Dehghani, M. Marx, and J. Kamps, "Time-aware authorship attribution for short text streams," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2015, pp. 727–730.

[114] S. R. Boutwell, "Authorship attribution of short messages using multimodal features," DTIC Document, Tech. Rep., 2011.

[115] R. Farahbakhsh, Á. Cuevas, and N. Crespi, "Characterization of cross-posting activity for professional users across facebook, twitter and google+," *Social Network Analysis and Mining*, vol. 6, no. 1, pp. 1–14, 2016.

[116] G. K. Mikros and K. Perifanos, "Authorship attribution in greek tweets using author's multilevel n-gram profiles." in *AAAI Spring Symposium: Analyzing Microtext*, 2013.

[117] W. X. Zhao, J. Jiang, J. Weng, J. He, E.-P. Lim, H. Yan, and X. Li, "Comparing twitter and traditional media using topic models," in *European Conference on Information Retrieval*. Springer, 2011, pp. 338–349.

[118] L. Hong and B. D. Davison, "Empirical study of topic modeling in twitter," in *Proceedings of the first workshop on social media analytics*. ACM, 2010, pp. 80–88.

[119] Y. Seroussi, F. Bohnert, and I. Zukerman, "Authorship attribution with author-aware topic models," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics, 2012, pp. 264–269.

[120] E. Stamatatos, "A survey of modern authorship attribution methods," *Journal of the American Society for information Science and Technology*, vol. 60, no. 3, pp. 538–556, 2009.

[121] J. A. Horne and O. Ostberg, "A self-assessment questionnaire to determine morningness-eveningness in human circadian rhythms." *International journal of chronobiology*, vol. 4, no. 2, pp. 97–110, 1975.

[122] Pew, "News use across social media platforms 2018," http://www.journalism.org/2018/09/10/news-use-across-social-media-platforms-2018, 2018.

[123] A. E. Holton, M. Coddington, S. C. Lewis, and H. G. De Zuniga, "Reciprocity and the news: The role of personal and social media reciprocity in news creation and consumption," *International Journal of Communication*, vol. 9, p. 22, 2015.

[124] A. Ju, S. H. Jeong, and H. I. Chyi, "Will social media save newspapers? examining the effectiveness of facebook and twitter as news platforms," *Journalism Practice*, vol. 8, no. 1, pp. 1–17, 2014.

[125] R. Hladík and V. Štětka, "The powers that tweet: Social media as news sources in the czech republic," *Journalism Studies*, vol. 18, no. 2, pp. 154–174, 2017.

[126] P. Rajapaksha, R. Farahbakhsh, and N. Crespi, "Identifying content originator in social networks," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[127] N. F. at Harvard, "20 years ago today, nytimes.com debuted "on-line" on the web," http://www.niemanlab.org/2016/01/20-years-ago-today-nytimes-com-debuted-on-line-on-the-web, 2018.

[128] G. News and M. Limited, "Guardian to offer news online first," https://www.theguardian.com/media/2006/jun/07/theguardian.pressandpublishing, 2018.

[129] A. Hermida, F. Fletcher, D. Korell, and D. Logan, "Share, like, recommend: Decoding the social media news consumer," *Journalism studies*, vol. 13, no. 5-6, pp. 815–824, 2012.

[130] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 591–600.

[131] A. O. Larsson, "The news user on social media: A comparative study of interacting with media organizations on facebook and instagram," *Journalism studies*, vol. 19, no. 15, pp. 2225–2242, 2018.

[132] M. Osborne and M. Dredze, "Facebook, twitter and google plus for breaking news: Is there a winner?" in *Eighth International AAAI Conference on Weblogs and Social Media*, 2014.

[133] C. Castillo, M. El-Haddad, J. Pfeffer, and M. Stempeck, "Characterizing the life cycle of online news stories using social media reactions," in *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, 2014, pp. 211–223.

[134] R. Bandari, S. Asur, and B. A. Huberman, "The pulse of news in social media: Forecasting popularity," in *Sixth International AAAI Conference on Weblogs and Social Media*, 2012.

[135] B. Yu, M. Chen, and L. Kwok, "Toward predicting popularity of social marketing messages," in *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*. Springer, 2011, pp. 317–324.

[136] C. Orellana-Rodriguez, D. Greene, and M. T. Keane, "Spreading the news: how can journalists gain more engagement for their tweets?" in *Proceedings of the 8th ACM Conference on Web Science*, 2016, pp. 107–116.

[137] ——, "Spreading one's tweets: How can journalists gain attention for their tweeted news?" *The Journal of Web Science*, vol. 3, 2017.

[138] A. Olteanu, C. Castillo, N. Diakopoulos, and K. Aberer, "Comparing events coverage in online news and social media: The case of climate change," in *Ninth International AAAI Conference on Web and Social Media*, 2015.

[139] L. McCay-Peet, M. Lalmas, and V. Navalpakkam, "On saliency, affect and focused attention," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012, pp. 541–550.

[140] J. Lehmann, C. Castillo, M. Lalmas, and E. Zuckerman, "Transient news crowds in social media," in *Seventh International AAAI Conference on Weblogs and Social Media*, 2013.

[141] D. M. Romero, W. Galuba, S. Asur, and B. A. Huberman, "Influence and passivity in social media," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2011, pp. 18–33.

[142] C. Orellana-Rodriguez, D. Greene, and M. T. Keane, "Spreading one's tweets: How can journalists gain attention for their tweeted news?" *The Journal of Web Science*, vol. 3, 2017.

[143] ——, "Spreading one's tweets: How can journalists gain attention for their tweeted news?" *The Journal of Web Science*, vol. 3, 2017.

[144] P. Rajapaksha, R. Farahbakhsh, and N. Crespi, "Identifying content originator in social networks," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[145] S. Shehnepoor, M. Salehi, R. Farahbakhsh, and N. Crespi, "Netspam: A network-based spam detection framework for reviews in online social media," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 7, pp. 1585–1595, 2017.

[146] S. Ozawa, S. Yoshida, J. Kitazono, T. Sugawara, and T. Haga, "A sentiment polarity prediction model using transfer learning and its application to sns flaming event detection," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2016, pp. 1–7.

[147] S. Yoshida, J. Kitazono, S. Ozawa, T. Sugawara, T. Haga, and S. Nakamura, "Sentiment analysis for various sns media using naïve bayes classifier and its application to flaming detection," in *2014 IEEE Symposium on Computational Intelligence in Big Data (CIBD)*. IEEE, 2014, pp. 1–6.

[148] P. J. Moor, A. Heuvelman, and R. Verleur, "Flaming on youtube," *Computers in human behavior*, vol. 26, no. 6, pp. 1536–1546, 2010.

[149] A. K. Turnage, "Email flaming behaviors and organizational conflict," *Journal of Computer-Mediated Communication*, vol. 13, no. 1, pp. 43–59, 2007.

[150] F. Mungeam and H. Crandall, "Commenting on the news: How the degree of anonymity affects flaming online," *Unpublished thesis, MA Program in Communication and Leadership Studies, Gonzaga University*, pp. 1–38, 2011.

[151] F. H. Khan, U. Qamar, and S. Bashir, "A semi-supervised approach to sentiment analysis using revised sentiment strength based on sentiwordnet," *Knowledge and information Systems*, vol. 51, no. 3, pp. 851–872, 2017.

[152] A. Gatt and E. Krahmer, "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation," *Journal of Artificial Intelligence Research*, vol. 61, pp. 65–170, 2018.

[153] A. H. Ombabi, O. Lazzez, W. Ouarda, and A. M. Alimi, "Deep learning framework based on word2vec and cnn-for users interests classification," in *2017 Sudan Conference on Computer Science and Information Technology (SCCSIT)*. IEEE, 2017, pp. 1–7.

[154] X. Ge, X. Jin, and Y. Xu, "Research on sentiment analysis of multiple classifiers based on word2vec," in *2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 2. IEEE, 2018, pp. 230–234.

[155] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1253, 2018.

[156] M. U. Salur and İ. Aydin, "Sentiment classification based on deep learning," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2018, pp. 1–4.

[157] X. Wang, W. Jiang, and Z. Luo, "Combination of convolutional and recurrent neural network for sentiment analysis of short texts," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2428–2437.

[158] C. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 69–78.

[159] X. Wang, Y. Liu, S. Chengjie, B. Wang, and X. Wang, "Predicting polarities of tweets by composing word embeddings with long short-term memory," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, 2015, pp. 1343–1353.

[160] Y. Liu, J.-W. Bi, and Z.-P. Fan, "Multi-class sentiment classification: The experimental comparisons of feature selection and machine learning algorithms," *Expert Systems with Applications*, vol. 80, pp. 323–339, 2017.

[161] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[162] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext. zip: Compressing text classification models," *arXiv preprint arXiv:1612.03651*, 2016.

[163] S. Kiritchenko and S. M. Mohammad, "The effect of negators, modals, and degree adverbs on sentiment composition," *arXiv preprint arXiv:1712.01794*, 2017.

[164] C. H. E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in *Eighth International Conference on Weblogs and Social Media (ICWSM-14). Available at (20/04/16) http://comp. social. gatech. edu/papers/icwsm14. vader. hutto. pdf*, 2014.

[165] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," *CS224N Project Report, Stanford*, vol. 1, no. 12, 2009.

[166] A. Anand, T. Chakraborty, and N. Park, "We used neural networks to detect clickbaits: You won't believe what happened next!" in *European Conference on Information Retrieval*. Springer, 2017, pp. 541–547.

[167] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine learning*, vol. 39, no. 2-3, pp. 103–134, 2000.

[168] B. Sharp and C. Chibelushi, "Text segmentation of spoken meeting transcripts," *International Journal of Speech Technology*, vol. 11, no. 3-4, p. 157, 2008.

[169] T.-C. Huang, C.-H. Hsieh, and H.-C. Wang, "Automatic meeting summarization and topic detection system," *Data Technologies and Applications*, 2018.

[170] F. Liu, F. Liu, and Y. Liu, "Automatic keyword extraction for the meeting corpus using supervised approach and bigram expansion," in *2008 IEEE Spoken Language Technology Workshop*. IEEE, 2008, pp. 181–184.

[171] ——, "A supervised framework for keyword extraction from meeting transcripts," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 3, pp. 538–548, 2010.

[172] T. Nakanishi, R. Okada, Y. Tanaka, Y. Ogasawara, and K. Ohashi, "A topic extraction method on the flow of conversation in meetings," in *2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)*. IEEE, 2017, pp. 351–356.

[173] M. Purver, T. L. Griffiths, K. P. Körding, and J. B. Tenenbaum, "Unsupervised topic modelling for multi-party spoken discourse," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 17–24.

[174] D. Gunawan, A. Pasaribu, R. Rahmat, and R. Budiarto, "Automatic text summarization for indonesian language using textteaser," *Proceeding of the Electrical Engineering Computer Science and Informatics*, vol. 3, no. 1, p. 012048, 2016.

[175] M. H. Bokaei, H. Sameti, and Y. Liu, "Summarizing meeting transcripts based on functional segmentation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 10, pp. 1831–1841, 2016.

[176] S. Banerjee, P. Mitra, and K. Sugiyama, "Generating abstractive summaries from meeting transcripts," in *Proceedings of the 2015 ACM Symposium on Document Engineering*, 2015, pp. 51–60.

[177] H.-J. Song, J. Go, S.-B. Park, S.-Y. Park, and K. Y. Kim, "A just-in-time keyword extraction from meeting transcripts using temporal and participant information," *Journal of Intelligent Information Systems*, vol. 48, no. 1, pp. 117–140, 2017.

[178] G. Tur, A. Stolcke, L. Voss, S. Peters, D. Hakkani-Tur, J. Dowding, B. Favre, R. Fernández, M. Frampton, M. Frandsen, *et al.*, "The calo meeting assistant system," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1601–1611, 2010.

[179] S. Huang and S. Renals, "Unsupervised language model adaptation based on topic and role information in multiparty meetings," in *Ninth Annual Conference of the International Speech Communication Association*, 2008.

[180] Y. Ha, J. Kim, D. Won, M. Cha, and J. Joo, "Characterizing clickbaits on instagram," in *Twelfth International AAAI Conference on Web and Social Media*, 2018.

[181] J. Zhu, K. Wang, Y. Wu, Z. Hu, and H. Wang, "Mining user-aware rare sequential topic patterns in document streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1790–1804, 2016.

[182] D. M. Blei, "Probabilistic topic models," *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.

[183] S. Dennis, T. Landauer, W. Kintsch, and J. Quesada, "Introduction to latent semantic analysis," in *25th Annual Meeting of the Cognitive Science Society. Boston, Mass*, 2003, p. 25.

[184] T. Hofmann, "Probabilistic latent semantic analysis," *arXiv preprint arXiv:1301.6705*, 2013.

[185] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[186] M. Vijaymeena and K. Kavitha, "A survey on similarity measures in text mining," *Machine Learning and Applications: An International Journal*, vol. 3, no. 2, pp. 19–28, 2016.

[187] N. Nakatsu, Y. Kambayashi, and S. Yajima, "A longest common subsequence algorithm suitable for similar text strings," *Acta Informatica*, vol. 18, no. 2, pp. 171–179, 1982.

[188] S. Niwattanakul, J. Singthongchai, E. Naenudorn, and S. Wanapu, "Using of jaccard coefficient for keywords similarity," in *Proceedings of the international multiconference of engineers and computer scientists*, vol. 1, no. 6, 2013, pp. 380–384.

[189] S. T. Dumais, "Latent semantic analysis," *Annual review of information science and technology*, vol. 38, no. 1, pp. 188–230, 2004.

[190] E. Gabrilovich, S. Markovitch, *et al.*, "Computing semantic relatedness using wikipedia-based explicit semantic analysis." in *IJcAI*, vol. 7, 2007, pp. 1606–1611.

[191] A. Joshi, P. Bhattacharyya, and M. J. Carman, "Automatic sarcasm detection: A survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 5, pp. 1–22, 2017.

[192] A. Zadeh, M. Chen, S. Poria, E. Cambria, and L.-P. Morency, "Tensor fusion network for multimodal sentiment analysis," *arXiv preprint arXiv:1707.07250*, 2017.

[193] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[194] P. Thomas, "Clickbait identification using neural networks," *arXiv preprint arXiv:1710.08721*, 2017.

[195] GLUE, "The general language understanding evaluation (glue) benchmark," https://gluebenchmark.com/leaderboard, 2020.

[196] P. Rajapaksha, R. Farahbakhsh, and N. Crespi, "Scrutinizing news media cooperation in facebook and twitter," *IEEE Access*, 2019.

[197] P. Rajapaksha, R. Farahbakhsh, N. Crespi, and B. Defude, "Inspecting interactions: Online news media synergies in social media," in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2018, pp. 535–539.

[198] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[199] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.

[200] E. Voita, R. Sennrich, and I. Titov, "The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives," *arXiv preprint arXiv:1909.01380*, 2019.

[201] E. Voita, P. Serdyukov, R. Sennrich, and I. Titov, "Context-aware neural machine translation learns anaphora resolution," *arXiv preprint arXiv:1805.10163*, 2018.

[202] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," *arXiv preprint arXiv:1905.09418*, 2019.

[203] P. Ganesh, Y. Chen, X. Lou, M. A. Khan, Y. Yang, D. Chen, M. Winslett, H. Sajjad, and P. Nakov, "Compressing large-scale transformer-based models: A case study on bert," *arXiv preprint arXiv:2002.11985*, 2020.

[204] M. A. Gordon, K. Duh, and N. Andrews, "Compressing bert: Studying the effects of weight pruning on transfer learning," *arXiv preprint arXiv:2002.08307*, 2020.

[205] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[206] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for bert model compression," *arXiv preprint arXiv:1908.09355*, 2019.

[207] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "Tinybert: Distilling bert for natural language understanding," *arXiv preprint arXiv:1909.10351*, 2019.

[208] R. Tang, Y. Lu, L. Liu, L. Mou, O. Vechtomova, and J. Lin, "Distilling task-specific knowledge from bert into simple neural networks," *arXiv preprint arXiv:1903.12136*, 2019.

[209] Y. Mao, Y. Wang, C. Wu, C. Zhang, Y. Wang, Y. Yang, Q. Zhang, Y. Tong, and J. Bai, "Ladabert: Lightweight adaptation of bert through hybrid model compression," *arXiv preprint arXiv:2004.04124*, 2020.

[210] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, "Mobilebert: a compact task-agnostic bert for resource-limited devices," *arXiv preprint arXiv:2004.02984*, 2020.

[211] F.-M. Guo, S. Liu, F. S. Mungall, X. Lin, and Y. Wang, "Reweighted proximal pruning for large-scale language representation," *arXiv preprint arXiv:1909.12486*, 2019.

[212] A. Fan, E. Grave, and A. Joulin, "Reducing transformer depth on demand with structured dropout," *arXiv preprint arXiv:1909.11556*, 2019.

[213] O. Kovaleva, A. Romanov, A. Rogers, and A. Rumshisky, "Revealing the dark secrets of bert," *arXiv preprint arXiv:1908.08593*, 2019.

[214] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does bert look at? an analysis of bert's attention," *arXiv preprint arXiv:1906.04341*, 2019.

[215] S. Jain and B. C. Wallace, "Attention is not explanation," *arXiv preprint arXiv:1902.10186*, 2019.

[216] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in nlp," *arXiv preprint arXiv:1906.02243*, 2019.

[217] R. Futrzynski, "Getting meaning from text: self-attention step-by-step," https://peltarion.com/blog/data-science/self-attention-video, 2020.

[218] Huggingface, "Bert," https://huggingface.co/transformers/model_doc/bert.html, 2020.

[219] P. Rajapaksha, R. Farahbakhsh, and N. Crespi, "Scrutinizing news media cooperation in facebook and twitter," *IEEE Access*, 2019.

[220] P. Rajapaksha, R. Farahbakhsh, N. Crespi, and B. Defude, "Uncovering flaming events on news media in social media," in *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*. IEEE, 2019, pp. 1–8.

List of figures

List of tables