
[All ETDs from UAB](#)

[UAB Theses & Dissertations](#)

2022

Adapting NLP Techniques in Transfer Learning For Forensic Authorship Profiling

Esam Alzahrani
University Of Alabama At Birmingham

Follow this and additional works at: <https://digitalcommons.library.uab.edu/etd-collection>



Part of the [Engineering Commons](#)

Recommended Citation

Alzahrani, Esam, "Adapting NLP Techniques in Transfer Learning For Forensic Authorship Profiling" (2022). *All ETDs from UAB*. 3.
<https://digitalcommons.library.uab.edu/etd-collection/3>

This content has been accepted for inclusion by an authorized administrator of the UAB Digital Commons, and is provided as a free open access item. All inquiries regarding this item or the UAB Digital Commons should be directed to the [UAB Libraries Office of Scholarly Communication](#).

ADAPTING NLP TECHNIQUES IN TRANSFER LEARNING FOR FORENSIC
AUTHORSHIP PROFILING

by

ESAM ALZHRANI

LEON JOLOLIAN, COMMITTEE CHAIR
MOHAMMAD HAIDER
KARTHIKEYAN LINGASUBRAMANIAN
MURAT M. TANIK
EARL WELLS

A DISSERTATION

Submitted to the graduate faculty of The University of Alabama at Birmingham,
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

BIRMINGHAM, ALABAMA

2022

ADAPTING NLP TECHNIQUES IN TRANSFER LEARNING FOR FORENSIC AUTHORSHIP PROFILING

ESAM ALZAHRANI

COMPUTER ENGINEERING

ABSTRACT

Cybercrimes have risen and caused threats to regular internet users. In recent times, the increased use of online social networks (OSNs) allows people to easily share opinions, personal information, and others. Since a major part of OSNs content is textual content, immense research has focused on text analysis techniques using machine learning and Natural Language Processing (NLP). One important area of research focused on text analysis using machine learning is forensic text analysis. Digital forensics is a discipline concerns finding, preserving, and presenting admissible evidence in court. Sadly, the convenience of OSNs creates an optimal venue for cybercriminals to perform malicious activities. As observed, anonymous texts have been associated with suspicious activities; thus, techniques for deanonymization have been a focal research interest in the past years. Forensic authorship profiling or characterization is one area of interest that needs to be furtherly investigated on account of directing the course of the cybercrimes' investigation. Mostly, the techniques of authorship profiling are based on machine and deep learning techniques. Such techniques use stylometric or statistical features to build the models. Several components affect the quality of such techniques e.g., dataset size and quality, preprocessing techniques, features selection, and classification methods. Lately, a new promising technique has emerged in NLP, known as a transformer which effectively enabled transfer learning. In transfer learning, a model that has been trained on a general domain dataset can be reapplied to similar or different specific tasks. Transfer learning is

relatively an older technique in another field like computer vision, but it has recently been widely applied to many NLP tasks and showed astonishing performance. As a result, we chose to examine the application of transfer learning techniques to tackle profiling the age and gender of the author.

After an extensive review of authorship attribution and profiling in the past ten years, we have noticed some gaps in the field of forensic authorship profiling that need to be addressed. Currently, the proposed techniques in authorship profiling have some serious limitations in terms of the quality and size of examined datasets. Moreover, the current techniques face serious issues at larger scales. Another limitation we observed is that the proposed methods are mostly based on machine learning methods which sequentially are based on preprocessing techniques and feature engineering. In our study, we offer a thorough literature review that covers different methods and their evaluations and limitations. Typically, most machine and deep learning models go through the same phases of text preprocessing, features extraction, features selections, and model training; thus, we utilized the recently trendy technique of transfer learning, which is considered features-independent, to profile anonymous authors by revealing authors' characteristics using dataset from PAN authorship profiling tasks. By doing so, we examined the effect of the most used text-preprocessing techniques on profiling the age and gender of anonymous authors using the transfer learning technique with BERT as an example. In another case study, we compared BERT, RoBERTa, and BERTweet when used to categorize the age and gender of anonymous authors with recommended values of the selected models' hyperparameters to recognize the association of these values with overall performances of the model. Experimentally, we tested the impact of text tokenization in transfer learning

using BERT tokenizer, WordPiece, as an example and how a well-known issue such as out of vocabulary limit the interpretation of BERT's tokenizer. As a result, we utilized different techniques such as text enrichment, missing words and emojis dictionaries to mitigate the effect of text misrepresentation limitations.

Keywords: digital forensics, natural language processing, forensic text analysis, authorship profiling, artificial intelligence, cybercrime, transfer learning.

DEDICATION

To my parents, Shams, Abdullah, Mohammed, my siblings, my friends, my colleagues,
and Laura.

ACKNOWLEDGMENT

In the name of Allah, the most gracious, the most merciful, I express my ultimate gratitude and thank to Allah for giving me the power, patience, and perseverance to carry out this comprehensive study.

I would like to thank my supervisor, Prof. Leon Jololian, for his constant help, support, and encouragement. During this time, I did not only grow as a person, I also grew intellectually and professionally. As the prophet of Islam, peace be upon him, said:” **Whoever does not thank people has not thanked Allah.**” I gratefully acknowledge the help and expertise of my committee members, Dr. Karthikeyan Lingasubramanian, Dr. Mohammad Haider, Prof. Murat M. Tanik, and Prof. Earl Wells. I am thrilled to have had the chance to work with such a wonderful team. I am extremely grateful for their help and support throughout my research.

As parts of this dissertation were published, I would like to thank all the reviewers who shared their views and feedback about our work. Their comments and guidance helped to improve the overall quality of this work. Finally, I would like to thank my colleagues who indirectly or directly helped me to carry out this research through their constructive criticism and fruitful discussion

TABLE OF CONTENTS

ABSTRACT.....	i
DEDICATION.....	iv
ACKNOWLEDGMENT.....	v
LIST OF TABLES	ix
LIST OF FIGURES	x
LIST OF ABBREVIATIONS.....	xi
CHAPTER 1: INTRODUCTION.....	1
AI In Forensic Authorship Profiling And Attribution	5
Authorship Attribution And Profiling.....	5
Open-Set Problem.....	9
Text Representation	10
<i>ONE-HOT ENCODING</i>	12
<i>BAG OF WORDS (BoW)</i>	13
<i>N-Grams</i>	13
<i>Term Frequency And Inverse Document Frequency (Tf-Idf)</i>	14
<i>Word Embeddings</i>	15
Different Features Engineering And Modeling Techniques	
In Authorship Analysis	16
TRANSFER LEARNING.....	17
CHAPTER 2: LITERATURE REVIEW	26

Authorship Analysis.....	26
<i>Authorship Attribution</i>	26
<i>Authorship Verification</i>	28
<i>Authorship Profiling</i>	30
Text-Preprocessing	37
Feature Engineering	38
Unsolved Challenges	40
CHAPTER 3: METHODOLOGY	49
Traditional Vs Transfer Learning	52
Dataset.....	53
Experiments	58
Profiling Hate Speech Spreaders By Classifying Micro TextsUsing Bert Model (Case Study)(Esam & Leon, 2021)	63
Preprocessing Techniques Effect (Alzahrani & Jololian, 2021).....	66
Different PTMs Comparison (E Alzahrani et el., 2022).....	71
The Analysis of Bert Tokenizer	75
CHAPTER 4: RESULTS AND DISCUSSION.....	85
Fake News Spreaders Case Study (Esam & Leon, 2021).....	85
Text Pre-Processing Techniques Effect (Esam Alzahrani & Jololian, 2021)	87
The Comparative Study (E Alzahrani et al., 2022).....	89
The Analysis of Text Representation In Transfer Learning	91
CHAPTER 5: CONCLUSION	98
REFERENCES	101
APPENDIX:	

A Literature Review Features And Different Studies Comparison	115
B The Extended Version of Literature Review Studies Comparison	117
C Experiments Charts, Results, And Confusion Matrices	125
D The Frequency of Emojis In PAN 2017 Dataset.....	140

LIST OF TABLES

<i>Tables</i>	<i>Page</i>
Table 1. A summary of literature review (This is a short version, to see the full version see appendix A).....	43
Table 2. <i>The distribution of dataset per class</i>	54
Table 3. <i>The dataset statistics (The distribution of tweets per gender and the total number of authors)</i>	56
Table 4. <i>The most 10 frequent emojis in the dataset from PAN 2107</i>	57
Table 5. Examples preprocessing using regex in python.....	67
Table 6. Preprocessing cases.....	68
Table 7. Values of the hyperparameters for preprocessing effects	70
Table 8. Hyperparameters and values	72
Table 9. The suggested values hyperparameters.....	73
Table 10. The result of tokenization tweets before and after emojis translation	78
Table 11. Models settings for the achieved results	86
Table 12. Results of gender profiling experiments. We consider using cross-validation to test the accuracy of the built models. We split the dataset into 90% for training and 10% for testing.	88

Table 13. Results of age profiling experiments. We consider using cross-validation to test the accuracy of the built models. We split the dataset into 90% for training and 10% for testing.	88
Table 14. Experiment results for different hyperparameters	90
Table 15. The results of gender profiling after text normalization	97
Table 16 The variations of features	116
Table 17: The extended version of Literature review studies comparison	118

LIST OF FIGURES

<i>Figure</i>	<i>Page</i>
Figure 1. Author attribution or identification	7
Figure 2. Authorship profiling into a smaller group of people with the same attribute (e.g., age or gender)	8
Figure 3. An example of self-attention	20
Figure 4. Transformer architecture (Vaswani et al., 2017).....	22
Figure 5. The implementation of multi-headed attention and its subprocess, scaled dot- product attention (Vaswani et al., 2017).....	23
Figure 6. General implementation pipeline	51
Figure 7. The process knowledge transfer	53
Figure 8. Words count per tweet for the dataset from PAN 2016	55
Figure 9. Words count per tweet for the dataset from PAN 2017	56
Figure 10. Examples of PTMs	59
Figure 11. Examples of PTMs (The design of our proposed approach)	60
Figure 12. The 3 rd scenario dataset splitting	65
Figure 13. Experiments pipeline for preprocessing techniques study	69
Figure 14. Standard implementation steps to build a pre-trained model that profiles authors' ages and genders.	72
Figure 15. The result of text tokenization using different techniques	75
Figure 16. Out of Vocabulary mitigation model.....	77

Figure 17. The tokenization of the word “unfollowed” vs. “followed”	80
Figure 18. The tokenization of URLs, mentions, and tags by the BERT tokenizer	80
Figure 19. The effect of adding custom label	84
Figure 20. Vocabulary tokens length of PAN 17 dataset before emojis and emoticons translation.....	92
Figure 21. Vocabulary tokens length of PAN 17 dataset after emojis and emoticons translation.....	92
Figure 22. Sub-word tokens length of PAN 17 dataset before emojis and emoticons translation.....	93
Figure 23. Sub-word tokens length of PAN 17 dataset after emojis and emoticons translation.....	94
Figure 24. Vocabulary tokens length of PAN 17 dataset after emojis and emoticons translation and contractions expansion	94
Figure 25. Sub-word tokens length of PAN 17 dataset after emojis and emoticons translation and contractions expansion	95

LIST OF ABBREVIATIONS

AI: Artificial Intelligence

BERT: Bidirectional Encoder Representations from Transformers

BERTweet: A pre-trained language model for English Tweets

BMR: Bayesian Multinomial Regression

BoW: Bag of Words

BPE: Byte-Pair Encoding

EBG: Extended Brennan-Greenstadt

GLUE: General Language Understanding Evaluation

GPU: Graphical Processing Unit

HTML: HyperText Markup Language

LDA: Latent Dirichlet Allocation

LIWC: Linguistic Inquiry and Word Count

LRC: Latent-representation Contrastive

LSA: Latent semantic analysis

MAF: Morphological Annotation Framework

MRC: Machine Reading Comprehension

NLP: Natural Language Processing

NLTK: Natural Language Toolkit

OSN: Online Social Network

POS: Part of Speech

PTM: Pre-trained Models

RoBERTa: Robustly Optimized BERT Pretraining Approach

SCAP: Source Code Authorship Profiling

SQuAD v1.1: the Stanford Question Answering Dataset

SVM: Support Vector Machine

SWAG: Situations With Adversarial Generations

TF-IDF: Term Frequency and Inverse Document Frequency

URL: Uniform Resources Locator

CHAPTER 1

INTRODUCTION

Amid the tremendous development of artificial intelligence (AI) techniques in the past two decades, AI has come to be applied in a range of fields and specialties. As such, AI has become a sophisticated discipline by which other domains, including health care, the economy, and IT solutions, have advanced. Although the use of AI has not been widely adopted in digital forensics, which involves extracting, preserving, and presenting admissible digital evidence in courts, it has been established and increasingly experimented with (Rocha et al., 2017).

Although rapid changes in Internet technology have always posed new challenges to traditional techniques in digital forensics (Quick et al., 2014), the U.S. Department of Justice now ranks cybercrime as the greatest threat to U.S. national security, economic growth, and public safety (Yepes, Ray (ATX Forensics LLC, Austin, 2016). Indeed, cybercrime is regarded as the fastest-growing type of crime not only in the United States but also around the world (Yepes, Ray (ATX Forensics LLC, Austin, 2016). As concerns digital forensics, an important factor in investigations into any type of criminal activity is criminal profiling, in which investigators (i.e., profilers), by considering historical information, build suspect profiles containing demographic information (e.g., gender, age,

and geolocation). With such profiles, investigators can narrow down the number of suspects and direct the course of investigations to the most plausible candidates. Although traditional criminal profiling has received considerable attention from researchers and practitioners, cybercrime now imposes a need for new approaches and techniques that are able to support effective cybercrime profiling (Yepes, Ray (ATX Forensics LLC, Austin, 2016). As social media and other text-based platforms become an increasingly significant part of Internet use, forensic authorship profiling has received increasingly sustained attention (Colombini & Colella, 2011; Frantzeskou et al., 2006; Rocha et al., 2017; Stolerma et al., 2014; Warikoo, 2014). One technique that digital forensics researchers can use to develop criminal profiles is authorship analysis, which entails identifying, verifying, or profiling the most likely author(s) of a text without any prior knowledge of the target suspect, namely via approaches based on machine learning or NLP (Rocha et al., 2017). Authorship analysis was initially developed as a branch of linguistics in which linguists use stylometric, semantic, or syntactic analysis to infer the potential author or authors of a text. In turn, it became used by researchers in the humanities to authenticate disputed literary works such as Shakespeare's (Eder et al., 2013; Hoorn et al., 1999). As advances in computing power make methods of machine learning and NLP more appealing for authorship analysis, the development of techniques for deanonymizing authors has become essential.

Authorship analysis has three categories of practice—authorship attribution (i.e., identification), authorship verification, and authorship profiling (i.e., characterization)—that, despite differing in purpose, are remarkably similar in terms of techniques applied. Authorship attribution, used to identify the anonymous authors of texts, involves

techniques such as similarity measurement, machine learning, and stylometry. Using similar techniques, authorship verification, based on similarity detection, determines whether two pieces of text are written by the same author. By comparison, authorship profiling, which concerns retrieving anonymous authors' characteristics (e.g., gender, age, educational background, and country of origin) from texts, uses similar approaches used in authorship attribution and verification.

Of course, limitations have been observed in some methods of authorship analysis, including the lack of scaling with a larger dataset, bias toward the selected dataset, and obstacles posed by the brevity of text in today's communications. Short-text platforms such as microblogs on OSN pose a special challenge for authorship analysis, for their lack of detail challenges models to learn from texts of such short length and to create markers and indicators for their various authors. In turn, to address such setbacks posed by the use of brief versus long texts in communication today (Robert Layton et al., 2010), more advanced techniques in author profiling are needed.

To date, most problems in authorship analysis, especially in authorship attribution, have been addressed as closed-set problems. For that reason, no guarantee exists that the same solutions will achieve the same results in open-set problems (Koppel & Winter, 2013). Although differences between closed-set and open-set problems are detailed in Section 1.3, at base closed-set problems involve cases in which all candidate authors are included in the training dataset, and no other unknown authors are found during the cross-validation of the model. Open-set problems, by contrast, are trained similarly but can be tested with new unknown authors included. Of all three categories of authorship analysis, authorship profiling is considered to be an open-set problem, one in which the developed

method can be generalized and tested against new anonymous texts. According to (Koppel & Winter, 2013), due to the open-set problem, authorship verification is far more complicated than attribution, and the same applies to authorship profiling, which is implemented with the open-set assumption. Nevertheless, a clear relationship exists between forensic authorship attribution and authorship profiling, for both involve using text analysis to attribute or classify text as a means to identify and/or categorize authors (Rocha et al., 2017).

Compared with authorship attribution, authorship profiling has not received much attention from researchers. To be specific, authorship profiling involves analyzing anonymous texts to derive characteristics and demographic information about authors—or suspects, in forensics—including their gender, age, and level of education. Our extensive review of literature on authorship profiling revealed efficient, newly developed methods in NLP, including transfer learning, that have been proven to outperform many state-of-the-art techniques in executing NLP tasks. However, to the best of our knowledge, no study has involved applying transfer learning techniques in author profiling according to age and gender. In response, we propose this study to examine the feasibility of using transfer learning to perform that task. Relative to current methods used in authorship profiling, which suffer from poor accuracy and nonstandard methods in approaching the problem, transfer learning is a stable, systematic method that can be generalized and possibly standardized. At the same time, whereas current methods of author profiling, all largely based on features engineering, have spawned significant variation in each model used, transfer learning usually requires a preprocessed text to be fed into the model.

Considering the variations in potential preprocessing techniques, we propose to conduct an experimental study that involves applying the most common techniques to measure each of their effects while using three selected pretrained models: the bidirectional encoder representations from transformers model (BERT) model, chosen for being one of the most-used stock pretrained models; and the robustly optimized BERT pretraining approach (RoBERTa), chosen for being an optimized version of BERT that uses different batch sizes, different learning rates, and, most importantly, different tokenizers; BERTweet, based on BERT and differs in the technique of pretraining as it has been pretrained on large tweeter corpus. We also compared the three selected models for their effects while being used with different hyperparameters in order to observe their performance in profiling the age and gender of authors.

We believe that the proposed method will support a reliable systematic methodology for approaching forensic authorship profiling as a means to help investigators to reveal vital information about suspects and consequently (re)direct the course of investigations. Moreover, because authorship profiling is primarily used in larger datasets and can be efficiently scaled to big data platforms (e.g., Twitter, Facebook, and YouTube), our proposed method offers a comprehensive analytical study that allows criminal profiling by analyzing the texts of suspects using transfer learning techniques. By using transfer learning, we intend to examine the variable factors that are associated with the performance of such techniques in the domain of forensic authorship profiling aiming to organize and generalize the effort to establish authorship profiling models using transfer learning.

AI In Forensic Authorship Profiling And Attribution

With the ever-increasing popularity of OSNs, Internet users are increasingly inclined to share opinions, interests, and personal information in volumes that can be difficult to conceive. The analysis of such unstructured data is thus more valuable than ever before. Text analysis using AI techniques has been applied in various fields, including marketing and NLP; in marketing, for example, companies use sentiment analysis to measure customers' satisfaction with certain products (P. Yang & Chen, 2018). By contrast, forensic analysis for such data remains in its infancy, especially in the field of deanonymization (Halimi & Ayday, 2017). Nevertheless, because forensic authorship attribution involves attributing anonymous texts to authors for presentation as evidence in legal proceedings (Rocha et al., 2017), such evidence has to be precise and error-free. In reality, however, forensic authorship attribution is far from being able to furnish admissible or even reliable evidence in court (Rocha et al., 2017), and the same applies to forensic authorship profiling. The most popular AI methods used in NLP and text classification include machine learning methods e.g. support vector machines (SVM) , naive Bayes, maximum entropy, and deep learning methods e.g. artificial neural networks (P. Yang & Chen, 2018).

Authorship Attribution And Profiling

Although the primary purpose of OSNs, as their name suggests, is to socialize and exchange with other users, they may also be used for criminal activity. Due to the large scale of social media data, with the characteristics of all big data—velocity, volume, and variety (Rocha et al., 2017)—authorship attribution becomes a problematic technique that is not as efficient as it should be. Figure 1 shows the process of identifying one author from

n authors, where n is constrained to the maximum number of closed known authors. The process achieves better results within a closed-set assumption, which is beneficial with limited scenarios that do not represent real case scenarios, than an open-set assumption.

An alternative technique that can be used to direct forensic investigations and implement profiles for potential suspects is authorship profiling. In forensics, profiling was established by a former agent of the Federal Bureau of Investigation, John Edward Douglas (Yepes, Ray (ATX Forensics LLC, Austin, 2016). John began studying and analyzing serial killers' crimes and, in time, interviewed some of the serial killers to find patterns associated with their criminal activities and to profile their characteristics. Since then, the approach has been thought to help to direct the course of investigations toward identifying the most plausible suspects of crimes. Profiling has been used in a broad range of investigations and, at times, provided breakthroughs able to redirect the course of investigations. The case of the Unabomber, the serial terrorist, ranks among the most famous cases in which profiling proved to be an essential factor in identifying the criminal, namely by analyzing the manifestos that the suspect was publishing. As (Keretna et al., 2013) have shown, writing styles are indeed affected by factors including culture, educational background, and the environment in which authors are raised.

Because anonymity is frequently associated with criminal activity and because criminals are known to seek anonymity to avoid getting caught by law enforcement (Halimi & Ayday, 2017; Rocha et al., 2017), techniques are required to overcome the challenges that anonymous users and authors pose in cyberspace. Although most OSNs are regarded as auxiliary platforms in which users use their real names, the option of anonymity remains available. In turn, author profiling in OSNs can determine the identifying characteristics of

anonymous cybercriminals (Halimi & Ayday, 2017) and can even be automated by AI techniques. According to (Halimi & Ayday, 2017), if profile matching across different OSNs proves to be possible, then such data can be used to build profiles for suspects in any other context. Moreover, the attributes used to detect similarity among different profiles can be used to build profile clusters by which users' profiles can be assigned. For example, location information can be used to classify the targeted anonymous user's geolocation.

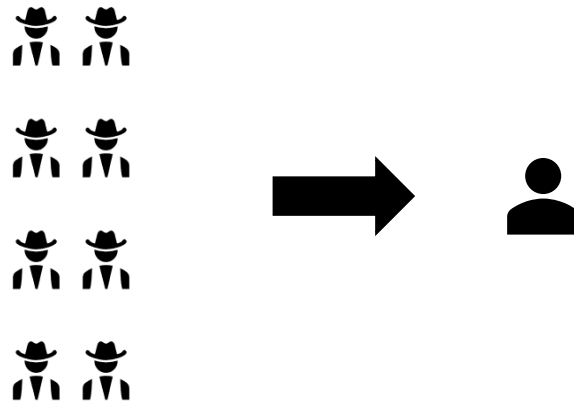


Figure 1. Author attribution or identification

Figure 2 shows the process of profiling in an open-set context by finding m users from n users, where m is a subcategory of n users that represents m users who share the same characteristic, in which $m < n$. Generally, m being subcategories for suspect characteristics and n being potentially unlimited. Attributes that can be examined by such profiling are general similarity, username similarity, location similarity, gender similarity, photo similarity, free text similarity, activity pattern similarity, interest similarity, and sentiment similarity (Halimi & Ayday, 2017). In the past decade, researchers began paying closer attention to the specific problem of identifying the age and gender of authors, and, as a result, efforts to optimize the methods of such identification have been tremendous.

However, given the recently developed method of transfer learning, the investigation of it and other methods could greatly benefit the advancement of authorship profiling.

In any study, the forensic analysis typically consist of four steps (Harichandran et al., 2016):

- *Identification of sources of evidence*, which will be concerned with identifying the various sources of data for the forensic investigation.
- *Collection and preservation*, which will entail capturing and preserving the data needed (e.g., text written by a suspect).

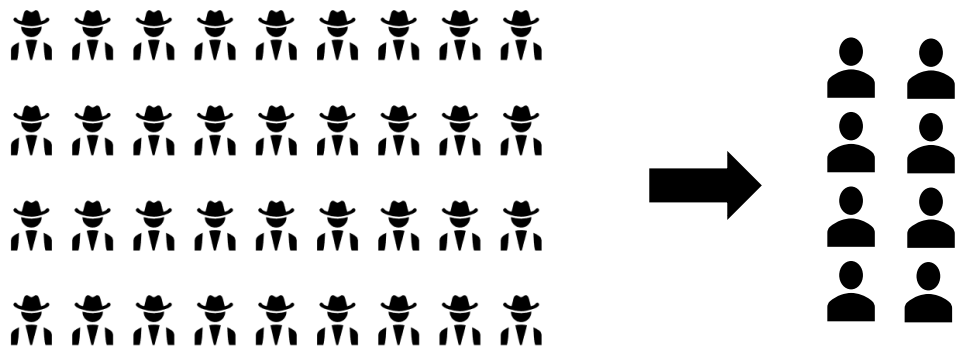


Figure 2. Authorship profiling into a smaller group of people with the same attribute (e.g., age or gender)

- *Examination and analysis*, which will involve analyzing and examining the forensic data.
- *Reporting and presentation*, which will be concerned with the legal presentation of all evidence obtained from the iterative processes.

Open-Set Problem

Validating machine learning models depends on the cross-validation method, in which the dataset is split into two folds; one-fold is used for training the model to learn to predict or classify, and the other fold is used to test the accuracy of the model. In authorship identification, if all authors are known in the dataset, and no unknown authors are present during training and testing, then the case is called a *closed-set case*. A closed-set case is not an optimal situation in real-life scenarios, for the model will be tested against any random text with no prior knowledge, a limitation that hinders the effectiveness of using such a mechanism. Moreover, the increase in the use of social media platforms with many millions of users renders such models useless. Recent research has overwhelmingly tended to show the results of models that were trained and tested in closed-set contexts where all users are known in both datasets (Stolerman et al., 2014). That tendency, however, presents a major challenge that prevents the field from advancement.

When Rocha et al. examined the behavior of authorship attribution models in an open-set case revealed that as the number of unknown authors increases, the accuracy of the models decreases noticeably (Rocha et al., 2017). However, because they included only 150 unknown authors and measured only the decline in accuracy, it remains unclear what kind of behavior can be expected when running the same models on massive platforms with millions of unknown users? We propose that authorship profiling will afford acceptable accuracy in both open- and closed-set cases. Nevertheless, as Koppel & Winter have stressed, solving a closed-set recognition problem does not guarantee solving an open-set problem (Koppel & Winter, 2013), meaning that the latter is considered to be the more challenging problem (Stolerman et al., 2014).

Text Representation

In the context of NLP, tokenization constitutes the process of transforming the text to unique numerical values. Tokenization is a mandatory step before using any AI models which can only handle numerical values. The examples and techniques of text tokenization are numerous. However, this study covers some of the most used techniques in NLP. The text representation started as simple as encoding the text to numerical values without any consideration to linguistic aspect. A token is described by Mielke et al. as "a relatively atomic word-like space-separated unit." Tokenization was formerly known as typographic units since it was motivated by linguistic aspects (Sabrina J. Mielke et al., 2021). Moreover, a token is described as "a non-empty contiguous sequence of graphemes or phonemes in a document" by the Morphological Annotation Framework (MAF)(Clément et al., 2010).

The Universal Dependencies guidelines referred to tokens and word-forms as "multi-tokens words" and "multiword tokens," respectively. According to Clement et al. and Sagot and Boullier, there is no one-to-one correspondence between tokens and word-forms (Clément et al., 2010; Sagot & Boullier, 2008); a word form can be made up of several word-forms and represented by the same token (for example, English don't = do + not) . The need for the atomic processing units (still known as tokens) to approximate word forms has diminished as a result of scientific findings (such as the impact of sub-words segmentation on machine translation performance) and technical requirements (such as language models like BERT that demand a fixed-size vocabulary) (Sennrich et al., 2016). Since tokenization is now a process without linguistic motivation, it is difficult to refer to typographic units as such. Instead, the former method that is linguistically based is now known as "pre-tokens," and the method of obtaining the tokens is known as "pre-

tokenization." Pre-tokenization is different from the new notion of tokenization in that it works with segmenting sentences at the sentence level rather than word level. The venerable Moses tokenizer (Koehn et al., 2007) and the more modern tokenizer package in HuggingFace's tokenization package are two of the most used tokenizers. Pre-tokenization can be used to normalize text, such as spelling correction, lemmatization, stemming, etc., in addition to producing tokenized units.

Because of the rarity of some terms, the lack of vocabulary increases. The problem is particularly severe in closed vocabulary models since they switch OOV for "UNK" tokens, which has the following negative effects:

- When performing natural language generation, UNKs are not allowed (NGL)
- They prevent us from extracting characteristics for novel words that are helpful anchors of meaning and not merely one-off events (Church, 2000) when utilized in large-scale models like ELMo (Peters et al., 2018) or BERT (Devlin et al., 2019)
 - It is impossible to eliminate rare words from languages other than English, especially ones with more productive morphology and larger type-token ratios (Cotterell et al., 2018; Sebastian J. Mielke et al., 2020)

However, there are new methods that handle uncommon vocabularies at the word level by putting the emphasis on the individual letters in a word. Pinter did a fantastic job of outlining the methods for better word representation (Pinter, 2021). By enhancing the BPE units' embedding with their constituent characters' embedding inspired by reducing text's noise with spelling errors or transferring to new domains like medical text, new techniques like CharaterBERT by EL Boukhouri et al. and CharBERT by Ma et al. used the same CNN construction as Kim et al. on new BERT-style models (El Boukkouri et al.,

2021; Kim et al., 2016; Ma et al., 2020). Next, Aguilar et al. consider similar technique but using transformer architecture as an alternative to CNNs (Aguilar et al., 2021). Pinter et al. present UNKs handler that simulates the embedding of a word given its spelling with the help of RNN model that is used if an unknown word is faced (Pinter, 2021).

ONE-HOT ENCODING

In one-hot encoding, each word w in the corpus vocabulary is assigned an integer ID (wid) that ranges from 1 to $|V|$, where V is the corpus vocabulary set. A V -dimensional binary vector of 0s and 1s is then used to represent each word. This is accomplished by filling a $|V|$ dimension vector with all 0s except the index, where $\text{index} = \text{wid}$. We just put a 1 in this index. Individual word representations are then concatenated to generate a sentence representation. For example, let $s = \text{BERT is a SOTA model}$ is a SOTA model. Therefore, each word will be assigned a unique ID: BERT = 1, is = 2, a = 3, SOTA = 4, model = 5. Each word will be represented in five-dimensional vector. The representation of the word BERT will be $[1\ 0\ 0\ 0\ 0]$ and it will be mapped to ID1.

To think of the scheme's advantages and disadvantages now that we have a better understanding of it. On the bright side, one-hot encoding is simple to comprehend and implement. It does, however, have a couple of flaws. A one-hot vector's size is proportional to the vocabulary's size, and most real-world corpora have a large number of vocabularies. This produces a sparse representation in which the majority of the vector items are zeroes, making it computationally inefficient to store, compute, and learn from (sparsity leads to overfitting). In addition, it does not have any consideration to the meaning or context.

BAG OF WORDS (BoW)

In BoW, the classification of different pieces of text is based on the occurrence of a word in a document. If two pieces of texts have nearly the same words, they will be added to the same group or bag despite the order or context. For example, let's say we have the following sentence: *'Mike likes to play basketball. Sam likes to watch movies';* the representation of its BoW will be as follows: {' ': 1, 'Mike': 1, 'Sam': 1, 'basketball': 1, 'likes': 2, 'movies': 1, 'play': 1, 'to': 2, 'watch': 1}. Clearly, BoW does not have consideration for context or order. As a result, it cannot recognize patterns of the occurrence of pair of words coming together e.g., the word likes comes before the word to in two instances.

N-Grams

As an alternative to the previously mentioned tokenization methods, n-gram has been proven to produce a better representation for text. It differs from the aforementioned methods i.e., one-hot-encoding and BoW because it recognizes the patterns of which n words or letters occur together. It is essential to mention that n-gram has two versions: word n-gram and letter n-gram. N-gram is based on a simple technique in which it divides the text into n-words/letters subgroups. Depending on the value of n, n-gram starts from the beginning of a piece of text and groups the words/letters in subgroups of size n until reaching the end of the text. For example, having the same sentence from the previous example *'Mike likes to play basketball. Sam likes to watch movies,* the word bigram will be as follows: [“Mike likes”, “likes to”, “to play”, “play basketball”, “Sam likes”, “likes to”, “to watch”, “watch movies”]. Now, we can see how n-gram can recognize patterns and have a better representation of a text as it recognizes the pair “likes to”, which occurred twice in the example.

Term Frequency And Inverse Document Frequency (Tf-Idf)

So far, all mentioned methods operate on the level of a single document and are isolated from the text structure and words frequency across all documents. To address that, TF-IDF works perfectly across multiple documents considering the words frequency and the occurrence of words per document to emphasize the importance of each word with every single document. Some words might have a high frequency in a document, but they might not be as frequent on other documents. This is why TF-IDF is a good candidate for text representation across multiple documents. To apply TF-IDF, there is a simple formula for Tf which measures the occurrence of a term in a given document. Because the lengths of the papers in the corpus vary, a term may appear more frequently in a longer text than in a shorter one. We divide the number of occurrences by the document's length to normalize the numbers. TF is defined as:

$$TF(t, d) = \frac{(\text{Number of occurrences of term } t \text{ in document } d)}{(\text{Total number of terms in the document } d)} \quad \text{Equation 1}$$

The term's relevance in a corpus is measured using IDF (inverse document frequency). All terms are given equal weight when computing TF (weightage). Stop words such as is, are, am, and others, on the other hand, are well-known for being unimportant, despite their widespread use. To adjust for these situations, IDF weights the terms that are relatively common across a corpus down and the rare terms up. The IDF of a phrase t is calculated in the following way:

$$IDF(t) = \log e \frac{(\text{Total number of document in the corpus})}{(\text{Number of documents with term } t \text{ in them})} \quad \text{Equation 2}$$

Word Embeddings

Embedding is another level of text representation that can capture "distributional similarities between words." Words that have similar meaning or lead to the same type can be recognized by embeddings. For example, other countries (e.g., Saudi Arabia, France, Canada, etc.) or cities in the United States could be distributionally comparable words if we're given the word "USA." If we are given the word "man," we can consider terms that have some relationship to it (e.g., father, son, king, etc.) to be distributionally comparable. These are words that are frequently used in similar situations. Word2vec is an example of words embedding which has been around since 2013 (Mikolov et al., 2013). Word2vec is a concept that takes a huge corpus of text as input and "learns" to represent the words in a common vector space depending on their contexts in the corpus. How do we determine the vector that best expresses the meaning of a word w and the words that exist in its context C ? We start with a vector \mathbf{v}_w initialized with random values for each word w in the corpus. Given the vectors for words in the context C , the Word2vec model refines the values in \mathbf{v}_w by predicting \mathbf{v}_w . A two-layer neural network is used to do this. We'll go into more detail on pre-trained embeddings before going on to creating our own. Recently, contextualized embedding has been introduced as an alternative to all previously mentioned methods. As the name suggests, in contextualized embedding the context of which a word is used matters. This is useful in the case of polysemy where the same word has different meaning based on the context it is used in. more details are covered in section of transfer learning.

Different Features Engineering And Modeling Techniques In Authorship Analysis

In NLP, function words can create extraneous information, because the everyday use of such words by most users decreases their value as markers of authors' styles. However, research has shown that function words can be used when coupled with another feature—for example, punctuation (Jurafsky & Martin, 2009). In term frequency-inverted document frequency (TF-IDF), commonly used for detecting similarity based on the analysis of the frequency of terms used across documents written by the same author (Jurafsky & Martin, 2009), unique words that are used by a specific author can be determined, especially after the elimination of function words. By contrast, the n -gram technique processes the sequence of terms that appear together throughout a given set of texts and thus gives an excellent picture of the probability of a word e given the word h , such that $p(e|h)$. The use of the n -gram technique affords the advantage of not needing linguistic knowledge (Rocha et al., 2017). Another technique is cross-topic or cross-genre attribution. Even though those techniques are not applicable for all scenarios due to the similarity of the topics in a dataset, they can be used in cases involving variant topics or genres.

As NLP becomes more sophisticated, it paves the way for the inclusion of complicated stylometric features to solve NLP tasks (Argamon et al., 2007)–(van Halteren, 2004). However, the use of such features is heavily dependent on the NLP toolkit and produces undesirable noise during the training of authorship attribution models (Rocha et al., 2017). In response, part of speech (POS) tagging is another feature that can be used as a marker for authors' writing styles. Beyond that, the complexity analysis of authorship signals is an alternative for machine learning and stylometric features; it uses the whole

text to quantify the signal of the text for later comparison with other signals in order to determine their similarity. In complexity analysis, the use of compression algorithms is a well-known technique (Oliveira et al., 2013).

TRANSFER LEARNING

Deep learning techniques, including widely used RNNs, CNNs, and GNNs, have greatly contributed to the improvement of NLP tasks in different applications. Although the use of deep learning overcomes some hurdles encountered in using machine learning techniques, the improvement in overall performance was not as significant as in the field of computer vision, the tasks in which benefited greatly from deep learning (Schaetti & Savoy, 2018). Unlike deep and transfer learning, traditional machine learning primarily relies on features of engineering efficiency, a limitation in which the feature handling of a target task plays an important role in the performance of the implemented model. That limitation that has been addressed by deep learning, however, which does not require the number of features needed by traditional machine learning. Even so, a problem that has been observed in both machine learning and deep learning techniques is the decline in performance when the model is applied to different data (Rocha et al., 2017). Such models do not generalize well and sometimes suffer from overfitting. In addition, the size of the data is another limitation faced in NLP in general. Deep learning networks, for example, use a relatively large number of parameters that might overfit small datasets.

Transfer learning has traditionally been connected with fine-tuning deep neural networks trained on the ImageNet dataset for use in other computer vision tasks (Malte & Ratadiya, 2019) . However, because of recent developments in natural language processing, transfer learning is now possible in this area as well. An aspect of next-

generation AI that has altered how problems in NLP are solved is pretrained models (PTM), which usually pretrain on large datasets and are reused for smaller ones in what are called *downstream tasks*. The technique reduces the time needed to manually label a large dataset to train a model from scratch. PTMs are trained on large unlabeled datasets (i.e., unsupervised learning) and transferred to labeled data tasks (i.e., semi-supervised or supervised). Xipeng divided pre-trained models into two generations: non-contextual language embedding and contextual language embedding (Qiu et al., 2020).

Over the years, the problem of sequence modeling received enormous attention from researcher. Subsequently, Recurrent neural networks have firmly established themselves as state-of-the-art method for sequence modeling and transduction issues like language modeling and machine translation—in particular, long short-term memory [13] and gated recurrent. Since then, other initiatives have continued to push the limits of encoder-decoder architectures and recurrent language models (Bahdanau et al., 2015; Cho et al., 2014; Chung et al., 2014; Hochreiter & Schmidhuber, 1997; Sutskever et al., 2014).

Recurrent models generally factor computation along the input and output sequences' symbol positions. They produce a series of hidden states h_t derived from a prior state h_{t-1} and fed to position t . This is done by aligning the positions to computation time steps. Due to memory limitations, batching across examples cannot be parallelized within training examples due to the inherent sequential character of the data, which is crucial for longer sequences. Through factorization techniques and conditional computation, recent work has significantly increased computational efficiency while also boosting model performance in the latter scenario (Kuchaiev & Ginsburg, 2019; Shazeer et al., 2017). Sequential computation continues to be subject to its fundamental limitation, nevertheless.

It is now possible to represent dependencies regardless of how far apart they are in the input or output sequences due to the incorporation of attention processes into powerful sequence modeling and transduction models in a variety of activities (Bahdanau et al., 2015; Kim et al., 2017). However, such attention processes are combined with a recurrent network in all but a few instances (Parikh et al., 2016).

Vaswani et al. present the Transformer, an architecture for a model that completely forgoes recurrence in favor of drawing global dependencies between input and output (Vaswani et al., 2017). The Transformer can achieve a new state of the art in translation quality after only twelve hours of training on eight P100 GPUs and offers substantially higher parallelization.

In order to construct a representation of the sequence, the attention mechanism known as "self-attention," also known as "intra-attention," links several points of a single sequence. Reading comprehension, abstractive summarization, textual entailment, and learning task-independent phrase representations are just a few of the tasks where self-attention has been used successfully (Cheng et al., 2016; Lin et al., 2017; Parikh et al., 2016; Paulus et al., 2018).

In order to determine self-attention, a query and a set of key-value pairs, all of which are vectors, are needed to calculate attention by multiplying the embedding by three trained matrices throughout the training phase, these vectors are produced. It is worth noting that these vectors are the embedding of each word which also happens to be the encoder's input vectors. The result is calculated as a weighted sum of the values, with each value's weight determined by the query's compatibility function with its corresponding key. Non-contextual embedding, as in skip-gram and GloVe, poses the disadvantage that the

model treats every word the same without any consideration of its context. That shortcoming affects the performance of embedding and the model overall because it provides semantic meaning only. By contrast, contextualized embedding pays attention to the reference and meaning of each word based on the context of the sentence in which it appears, or what is called *self-attention* (see Figure 3), which connects each word with other words in a sentence and computes a weighted score for each word. As a result, the weighted score creates a range of the most related words in association with the studied word to determine the meaning of the word in its specific context. As such, contextualized embedding has been the mainstream of recent research on NLP.

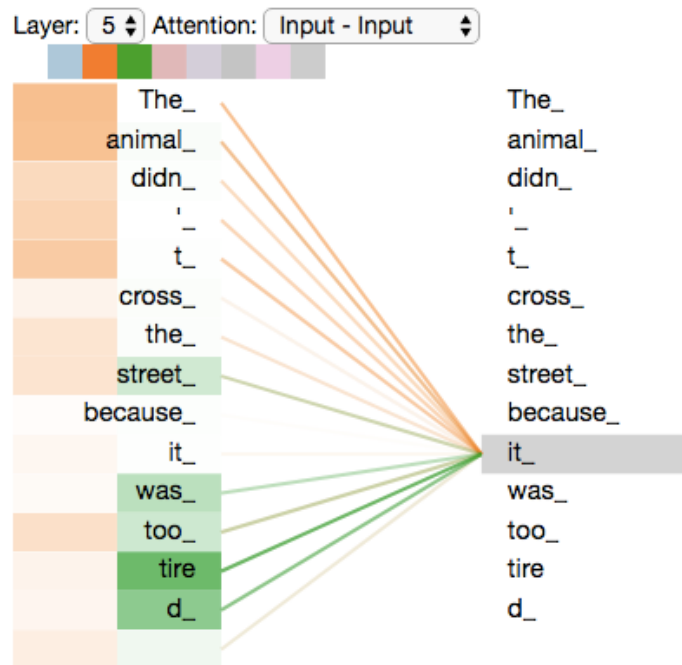


Figure 3. An example of self-attention

Although some contextual language PTMs are based on LSTM, most contextual PTMs are now based on the Transformer, a model architecture (see Figure 4) developed

by Google Brain and Google Research that is primarily based on the concept of self-attention, which substitutes the reliance on recurrence and convolution. Because the Transformer enables more parallelization, it substantially cuts the training time when compared with neural networks. The architecture of the Transformer consists of two primary parts—the encoder and decoder—stacked on top of each other and varying in the count based on the proposed model architecture’s variation. For instance, BERT-base has 12 layers, whereas BERT-large has 24 layers. The architecture of the encoder and decoder layers is nearly similar and depends on two sublayers: multi-headed self-attention and a position-wise feed-forward network. After each sublayer appear a residual connection and a normalization layer. The implementation of the Transformer has paved the way for the advent of second-generation PTMs.

The implementation of encoder and decoder differs in the masked self-attention on the decoder’s side. The decoder performs language modeling by predicting the next word in the sequence; thus, it has a masked multi-headed self-attention by which it predicts based on the previous sequence only, after which everything else is masked. For example, if a decoder is predicting a token at the i th position, then every token before the i th position is seen by the decoder, and every subsequent token is masked and cannot be seen by it.

As mentioned, self-attention is a function by which every token is given a weight associated with every other token’s weights and relies on three vectors: keys, values, and queries. Self-attention provides the weight of how much a token is related to all other tokens in a specific sequence.

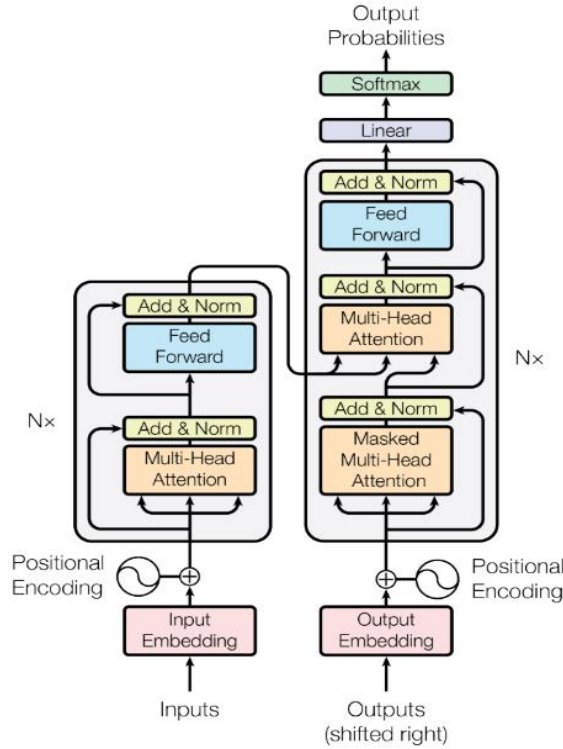


Figure 4. Transformer architecture (Vaswani et al., 2017)

That mechanism led to what is known as contextualized embedding, which differs from previous embedding techniques that could not provide semantic meaning for polysemous words based on the context in which they are used. The architecture of the self-attention layer is not especially complex, and it consists only of basic operations such as dot-product, scaling, and SoftMax (see Figure 5). Using d_{model} -dimensional keys, values, and queries, Vaswani et al. found it more advantageous to linearly project the keys, values, and queries h times with various learnt linear projections to d_k , d_k , and d_v dimensions, respectively. They then run the attention function in parallel on each of these projected copies of the queries, keys, and values to provide d_v -dimensional output values. The final results are shown in Figure 5 when these are combined and once more projected.

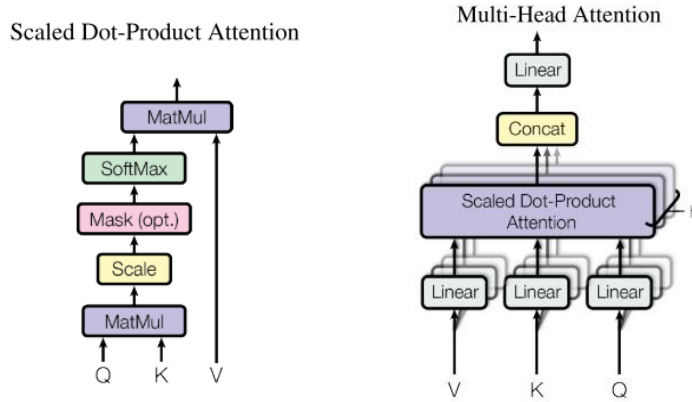


Figure 5. The implementation of multi-headed attention and its subprocess, scaled dot-product attention (Vaswani et al., 2017)

BERT is a prominent pre-trained model that utilizes the transformer architecture, which was released in 2017 (Devlin et al., 2019; Vaswani et al., 2017). It only utilizes the encoder component of the transformer in which text is tokenized based on the relation between each word and the rest of the text in multiple layers. The technique by which the contextualized tokenization has advanced is called self-attention. In each layer, the encoder will mask randomly a percentage of the text, nearly 15%. Since BERT was pre-trained on a large dataset, e.g., the content of English Wikipedia and a large corpus of more than 8,000 books, the model has an excellent ability to provide a contextualized tokenization for various contexts. One unique feature of BERT is the unidirectional nature of the model, in which the model considers texts in both directions to guarantee better representation of the text.

RoBERTa is based on BERT with few changes. According to the author of BERT, it needed more extended training (Liu et al., 2019). Thus, this is one of the optimizations in RoBERTa. They also removed the next sentence prediction along. The more extended sequence and dynamic language masking were considered in implementing RoBERTa.

BERTweet is a fusion between BERT and RoBERTa (Nguyen et al., 2020). It uses the architecture of BERT and the training procedure of RoBERTa.

BERT uses the WordPiece tokenizer to convert text into meaningful numerical values, and BERTweet utilizes an enhanced version of WordPiece tokenizer, which was trained on a large Twitter dataset from 2012–2020 (Devlin et al., 2019) (Liu et al., 2019; D. Q. Nguyen et al., 2020). RoBERTa employs byte-pair encoding (BPE) to encode the text; because words are represented in BPE by a combination of characters and word-levels, rather than deriving entire words through a statistical analysis of the training corpus, natural language corpora can accommodate immense vocabularies.

The purpose of this study is to analyze profiling the age and gender of the author if a pretrained model is used e.g., BERT, RoBERTa, and BERTweet. We provide a comprehensive analytical study in which multiple challenges will be tackled. The challenges we aim to investigate are the effect of different text-preprocessing techniques, the mitigation of out of vocabulary issues, and the generalization of applying transfer learning techniques in the domain of forensic authorship profiling. Many associated variables, which are author profiling specific, affect the overall performance of pretrained models. Thus, such variables were closely examined to provide general guidelines for applying pretrained models to profile the age and genders of authors. In the remainder of this document, the next section introduces past work related to author profiling, followed by a methodology section that details the implementation and different experimental methods we considered, and emphasizes the objective and steps performed in each experiment. After that, a section presents the results of each experiment and highlights the

impact of the model's accuracy in predicting the gender and age of authors. In the last section, we provided a summary and future direction.

CHAPTER 2

LITERATURE REVIEW

Authorship Analysis

Authorship Attribution

Authorship attribution is the most researched category and has received enormous attention from researchers compared to other authorship analysis subspecialties. Studies over the past two decades have provided important information on the different techniques that can be used to address authorship attribution challenges. The studies in this area of research vary in approaches and targeted datasets. Naïve Bayes is one of the simplest classification approaches and has been applied by many researchers (Clement & Sharp, 2003; Peng et al., 2004; Savoy, 2013; Zhao & Zobel, 2005). The same researchers used function words as their feature to train the classifier. Such a technique is straightforward and can lead to an acceptable accuracy.

In his analysis of the Federalist Articles, Savoy uses six different attribution schemes: Delta rule (F., 2002), Chi-Square metric (Grieve, 2007), KLD (Zhao & Zobel, 2007), Z-score method (Savoy, 2012), Naïve Bayes (Manning et al., 2008), and (SVM) (Joachims, 2002). Savoy suggests that using the voting method will result in the most accurate results when considering that every attribution scheme has the same importance (Savoy, 2012). Therefore, the prediction that has more counts will be the most probable

candidate. According to Zhao and Zobel, in a typical situation, documents with known authorship will be included in attribution training. Then, the same attribution will be used to attribute the anonymous documents or texts (Zhao & Zobel, 2005). They concluded that Bayesian networks are effective when used with function words as an only marker (Zhao & Zobel, 2005). Frantzeskou et al. and Layton et al. used source code authorship profiling (SCAP) with character n-gram as a feature (Frantzeskou et al., 2006, 2007; R Layton et al., 2012). This approach achieved a promising accuracy, proving the feasibility of using SCAP in authorship attribution tasks. Other researchers use SVM, the most common classifying approach in authorship attribution (Rocha et al., 2017).

According to Altamimi et al., SVM is found to be effective at large-scale and long text classification (Altamimi et al., 2019). Qian et al. provide a solution for a semi-supervised method with a limited labeled dataset (Qian et al., 2016). They propose a novel framework for authorship attribution learning, starting with limited training data. The method is considered a semi-supervised method, in which the model augments the training set based on the classifiers that were implemented in three views and inter-adding. They name this approach a tri-training method. The results show a better accuracy compared to baseline methods in semi-supervised learning. Stoleran et al. present a novel approach for authorship attribution by using classify-verify in an open-world setting (Stoleran et al., 2014). The method is based on an abstaining classifier in which the classifier, in some cases, refrains from classifications to reduce the error rate.

Two corpora were used in this paper; the first corpus is Extended Brennan-Greenstadt (EBG), and the second corpus is the ICWSM 2009 Spinn3r Blog dataset (Burton et al., 2009). Even though this approach is meant to deal with open-world problems

and performs well for that goal, it can be extended to closed-world settings to lower the error by refraining from attributing a document to the wrong author, leading to better accuracy overall. Random Forest classifiers are also among the classifiers that have been tried by some researchers

One of the limitations of authorship attribution arises when an author of the tested text was not included in the attribution training, the attribution of the unknown text will be falsely attributed to one of the authors who were included in the training of the attribution model. As stated by Zhao and Zobel, there is a limitation in the attribution papers caused by the variation in the selected features (Zhao & Zobel, 2005). Researchers do not take into account a systematic features selection, nor do they examine best candidate models for each specific authorship analysis task. Moreover, research on the subject has been mostly restricted to a limited small dataset and does not demonstrate the effectiveness of the proposed approach on large-scale datasets.

Authorship Verification

Authorship verification has received more attention since the PAN evaluation (Rocha et al., 2017). It is a 1:1 task to verify whether two documents have been written by the same person. The authorship verification is an open-set problem; thus, it greatly varies from closed-set authorship attribution. Koppel and Winter argue that "all standard closed-set authorship attribution solutions are reducible to the authorship verification problem, whereas the reverse is not true" (Koppel & Winter, 2013). The same researchers propose an imposters method, in which they generated imposter documents resembling what happens in a police lineup. The similarity measurement, along with the features' selection, will determine the best candidate. Using the imposters method, the researchers achieved

73.1% accuracy, which beats any baseline method that has been applied to authorship verification.

In the PAN 2015 authorship verification shared task, participants were asked to determine if a document was written by a given known author or not. Documents were given in four languages: English, Dutch, Spanish, and Greek. Pacheco et al. proposed a method based on a random forest features-encoding scheme based on three parts: (1) the complete set of authors, (2) the known authors, and (3) the target document to be tested for the verification task (Pacheco et al., 2015). They included many features such as statistics about the text, words and characters distributions, and linguistic features. They achieved second and third place in Dutch and Spanish datasets, respectively. Bartoli et al. introduced a machine learning method based on the differential of features for a different non-overlapping group of authors (Bartoli et al., 2015). They used multiple features to distinguish authors, such as word and character n-grams, POS, and some statistics about the text (e.g., sentence length, word length). To achieve the best performance, they used three different regressors: trees, random forest, and SVM. They ranked first in the Spanish language. Maitra et al. also proposed a method based on random forest, using features similar to those used by other teams (Maitra et al., 2015). They achieved third place in Dutch but did not perform very well in the other languages. Bagnall made use of Multi-headed Recurrent Neural Networks to mitigate the overfitting of using smaller corpora (Bagnall, 2015). The method achieved the first rank overall and first place in English and Greek in subcategories.

Authorship Profiling

In the effort to conduct a thorough survey for the literature most related to authorship profiling, we could not overlook the enormous effort of the PAN shared task in authorship profiling, which highlights the importance of the field. However, we have excluded some papers and results that are not related to the scope of our study. Papers that lack some details about the experiments were excluded. Besides, any task that is not focused on text analysis was also excluded. Finally, we limited our focus to English texts and excluded any non-English texts. As a result, we have studied the PAN authorship profiling shared tasks from 2013-17 (Rangel et al., n.d., 2013a, 2016, 2017; Rangel & Chugur, 2014). We named the dataset of each shared task as PAN and followed that by the last two digits of the year of the task. For example, if we want to refer to the dataset or any information from the shared task of 2014, it will be written as PAN14. In the following paragraphs, we will list a brief description of every task and the structure of the dataset.

In PAN13, participants were asked to categorize authors based on gender and age (Rangel et al., 2013b). The dataset was collected from netlog.com with the consideration of the variation of topics and the possibility of having automated generated texts (chatbots). The texts vary in length as short and long texts, making the data like the real-world scenario. The data are divided based on the age group of the authors. The age groups are divided into three groups: 10s (13-17), 20s (23-27), and 30s (33-47). The total number of training data is 236,600; 21,200 for early bird, and 25,440 authors for the test. There is another text dataset in Spanish, but we will only consider the English dataset for the sake of our study.

PAN14 also focused on gender and age profiling. It also considers the variation in genres, as it includes four different sub-corpora. The corpus is divided into four sub-corpora collected from social media, blogs, Twitter, and hotel reviews in both English and Spanish except for hotel reviews only in English. Each sub-corpora has a profile for each author within that corpus. Each author is labeled based on gender and age group as follows: a) 18-24, b) 25-34, c) 35-49, d) 50-64, e) 65+ (Rangel & Chugur, 2014).

Besides gender and age, participants in PAN15 were asked to identify the personality traits of Twitter users. The dataset was collected from Twitter in four languages: English, Spanish, Dutch, and Italian. All four language datasets were labeled for gender and personality traits. Only English and Spanish datasets were labeled for four age classes: a) 18-24, b) 25-34, c) 35-49, d) 50+ (Rangel et al., 2015). The personality trait sources were self-reported by using the BFI-10 online personality test (Rammstedt & John, 2007).

The PAN16 task investigated the effect of the cross-genres age and gender identification. The participants were asked to train their models on certain genres and test their models on a completely different genre. The datasets for training, early bird, and testing are from different sources to guarantee the corpora's variety. The corpus consists of different languages: English, Spanish, and Dutch. Each sub-corpus is labeled for age and gender, except the Dutch sub-corpus, which is only labeled for gender. The age subclasses are divided as follows: a) 18-24, b) 25-34, c) 35-49, d) 50-64, e) 65+ (Rangel et al., 2016).

The PAN17 task aimed to classify gender and language variety from four different languages: Arabic, Portuguese, Spanish, and English. There are variations (dialects) for each chosen language, so participants were asked to predict the dialect for each language

(Rangel et al., 2017). The variations included in this task are: (1) Arabic: Egypt, Gulf, Levantine, Maghrebi; (2) English: Australia, Canada, Great Britain, Ireland, New Zealand, United States; (3) Portuguese: Brazil, Portugal; (4) Spanish: Argentina, Chile, Colombia, Mexico, Peru, Spain, Venezuela. The dataset was collected from Twitter based on the region of each language variation.

SVM is the most common classification model for authorship analysis. We have observed it has been used by many researchers and comparatively, it has achieved a very good accuracy. Meina et al. and two other teams, Santosh et al. and Sapkota et al., chose to consider HTML and image URLs as features (Meina et al., 2013) (Santosh et al., 2013) (Sapkota & Solorio, 2013). Meina et al. ranked first in PAN13, achieving 64.9% and 59.2% accuracy in age and gender identification, respectively (Meina et al., 2013). One of the distinctive techniques Meina et al. considered is the classification of human-like and spam-like texts and the use of Naïve Bayes as a classifier (Meina et al., 2013). Santosh et al. and Sapkota et al. applied SVM as a classifier and achieved lower accuracy scores than Meina et al. (Santosh et al., 2013) (Sapkota & Solorio, 2013) (Meina et al., 2013).

Álvarez-Carmona et al. and González-gallardo et al. achieved the best accuracy for English text authorship profiling in PAN15 (Álvarez-Carmona et al., 2015; González-gallardo et al., 2015). Interestingly, they used a variety of features. Álvarez-Carmona et al. used Latent Semantic analysis with Second-order features combination based on the relationship among terms, documents, profiles, and sub-profiles. Further, González-gallardo et al. considered character n-gram and POS n-gram as their features (González-gallardo et al., 2015). They also applied a preprocessing technique by removing hashtags, URLs, and mentions. However, both teams used SVM (LibLinear) as a classifier for their

implementations. Grivas et al. used some content features like word length, letter case, and Twitter features (hashtags, links, mentions) (Grivas et al., 2015). They also used TF-IDF n-gram and SVM for age and gender classifiers. For the personality prediction, they used regression. Even though Posadas-Durán et al. used SVM (LibLinear) as their classifier, which achieved the best two accuracies for the other teams, the use of different features and preprocessing led to less accuracy by more than 20% when compared to Álvarez-Carmona et al. (Posadas-Durán et al., 2015) (Álvarez-Carmona et al., 2015; González-gallardo et al., 2015). Posadas-Durán et al. considered removing short tweets that contain less than five words and also considered using ten different types of n-gram like lemmas, words, relations, POS, and others (Posadas-Durán et al., 2015). Our best guess is that the use of different preprocessing techniques and features led to poor accuracy compared to the teams that ranked first and second.

The PAN16 task is quite different from previous tasks; it better demonstrates real-case scenarios by testing resulted models on different genres, which is typical in real life. There is no guarantee that the training dataset will be similar when a model is applied to random data. Expectedly, the first team Bougiatiotis & Krithara only achieved joint accuracy for age and gender of 39.74% (Bougiatiotis & Krithara, 2016). The first and second teams, Bougiatiotis and Krithara and Busger et al., used SVM as their classifier (Bougiatiotis & Krithara, 2016; Busger et al., 2016). In addition, they both used stylistic features (unique words, sentiment words, etc.) and Second-order representation as features. Unlike Busger et al., Bougiatiotis and Krithara applied preprocessing techniques by utilizing lemmatization, lowercase letters, number removal, and hashtag, RT, and URL removal. Modaresi et al. achieved the same joint accuracy as Busger et al. (Modaresi et al.,

2016)(Busger et al., 2016). However, Modaresi et al. used logistic regression as a classifier (Modaresi et al., 2016). Interestingly, they also used stylistic features (unique words, sentiment words, etc.).

In the PAN17, Basile et al. ranked first by using character and word n-grams and SVM as a classifier (Basile et al., 2017). Tellez et al. also used SVM classifier with only the bag-of-words feature and ranked second (Tellez et al., 2017). The best three teams, Basile et al., Tellez et al. , and Markov et al., used SVM with slight variations in the selected features (Basile et al., 2017), (Tellez et al., 2017), (Markov et al., 2017).

As observed, the first five submissions in PAN14 (López-Monroy et al., 2014), (Siang & Thing, 2014), (Maharjan et al., 2014), (Villena-román & González-cristóbal, 2014), (D. Weren et al., 2014) achieved acceptable results given the heterogeneity of the corpora. López-Monroy et al. proved that using many features without considering factors that affect the task at hand specifically will not always lead to better accuracy (López-Monroy et al., 2014). They considered studying the task specifically to come up with an architecture that better fits the target task. Therefore, the paper offers the use of intra-profile information to link the distinctive information with its corresponding class. Moreover, the classifier approach used is the standard LibLINEAR without any customization. Siang & Thing, Maharjan et al., and Villena-román & González-cristóbal, who ranked second, third, and fourth, respectively, all used n-grams and bag-of-words as features and logistic regression, except Villena-román & González-cristóbal, who used Multinomial Naïve Bayes (Siang & Thing, 2014) (Maharjan et al., 2014) (Villena-román & González-cristóbal, 2014).

Burger et al. collected a large dataset from Twitter, which contains more than 213 million tweets from 18.5 million users in different languages (Burger et al., 2011). Because of the large-scale dataset, manual annotation was infeasible. Instead, part of the dataset was manually annotated, and the rest of the dataset was annotated automatically using classifiers that were built based on the manually annotated dataset. This research aimed to characterize the gender of the author based on the author's text. Their experiments have classifiers built on one tweet text, multiple tweet texts, and multiple tweet texts with other fields (Screen name, Full name, description). The classifier, Balanced Winnow2 own implementation, which was built from all fields, outperformed others, with 98% accuracy on the development dataset and 91.8% on the test dataset. (D. Nguyen et al., 2013) analyzed tweets that were annotated manually to predict age in three different approaches: age categories, exact age, and life stage. The classification of age categories and life stage performed better than the prediction of the exact age when they used linear and logistic regressions for classification and unigram as their only feature.

Goswami et al. studied the stylometry variation based on gender and age (Goswami et al., 2009). They claimed that using two distinctive features, slang words and sentence length, will improve the overall accuracy of the implemented model. Their experiment uses Schler's corpus, which was collected in 2004 from blogger.com, and considers the Naïve Bayes classifier. They achieved a better accuracy of 80.32% using the augmented features than did Schler et al., who achieved 76.2% using content words (Schler et al., 2006). Mechti et al., on the other hand, considered multiple features such as the size of sentences, the occurrence of words, lexical analysis, and topic-related words, yet they ranked in last place (Mechti et al., 2014). Argamon et al. proposed an approach for text categorization for age,

gender, native language, and the “Big Five” personality dimensions (Argamon et al., 2009). The machine learning model, Bayesian Multinomial Regression (BMR), is trained on the processed data to create the classifier that will be later used for categorizing the unlabeled text. The feature used in this study is function words accompanied by part-of-speech for each function word. They also considered adding content-based features by including 1000 style marker words, which are believed to distinguish between variant writing styles accordingly with the categories of interest. The gender and age classification model performs better when content and style features are combined.

For researchers who use unusual techniques for authorship analysis such as deep learning approaches like RNN and CNN, distance-based approaches, and Exponential Gradient, Miura et al. ranked sixth in PAN17 as the best team who applied deep learning techniques (Miura et al., 2017). They selected a character- and word-embedding combination as a feature for the deep learning approach. Adame-Arcia et al. and Khan achieved poor accuracy of 0.1 and 0.19, respectively (Adame-Arcia et al., 2017) (Khan, 2017). It is worth mentioning that both used distance-based algorithms. (Przybyła & Teisseyre achieved the third-best accuracy in PAN15 by only using polarity words and emotions as features and distance-based approaches as their classifier (Przybyła & Teisseyre, 2015).

Koppel et al. offered an approach for categorizing authors’ gender using a genre-controlled corpus that was collected from a British National Corpus (Koppel et al., 2002). They started by considering a large set of features, including lexical and syntactic features. For example, a 405-function word list that appears at least once in the corpus has been considered by the researchers. They also used POS n-grams, including 76 parts-of-speech

tags. As a learning method, they used a variant of the Exponential Gradient algorithm. The researchers suggest that the genre of the documents affects the categorization directly. Therefore, they proposed controlling the genre of the document by training the model on only one genre as a prior step. After controlling the genre, they report an improvement in accuracy.

Text-Preprocessing

One step in NLP that is essential as a part of the implementation of models' pipeline. Some papers used uncommon preprocessing techniques e.g., extending shortened texts such as slang words, contractions, and abbreviations (Gómez-Adorno et al., 2016). Lundeqvist & Svensson removed HTML tags, and used Twitter custom tokenizer (nltk.tokenize package — NLTK 3.6.2 documentation) (Lundeqvist & Svensson, 2017). However, some papers considered common preprocessing techniques, similar to those used in PAN shared tasks (Mamgain et al., 2019). At least five research teams represented in PAN's shared tasks from 2013 to 2017 removed retweet tags from the texts during preprocessing (Rangel et al., 2013b, 2016, 2017, 2015; Rangel & Chugur, 2014); 17 groups removed hashtags (Rangel et al., 2013b, 2016, 2017, 2015; Rangel & Chugur, 2014), (Mamgain et al., 2019); and 19 teams considered removing URLs. For the removal of the mentioned tags, 17 research groups considered removing them from the processed text. Stop words were removed only four times (Bakkar Deyab et al., 2016; Kheng et al., 2017; Martinc et al., 2017) (Seelam et al., 2018), and 29 teams did not apply any preprocessing technique whatsoever (Rangel et al., 2013b, 2016, 2017, 2015; Rangel & Chugur, 2014). In 11 instances, retweet tags, URLs, and mentions were all removed (Rangel et al., 2013a, 2016, 2017, 2015; Rangel & Chugur, 2014).

The effectiveness of preprocessing techniques in machine learning approaches depends on the selection of features and classifiers. In transfer learning, the extensive training of pretrained models on large data equips them with the needed power to model the language and capture most of its contextualized aspects. Because transfer learning uses the previously learned knowledge to tokenize the downstream text (Panigrahi et al., 2021), it uses fine-tuning to train and classify the downstream task (Q. Yang et al., 2020).

Feature Engineering

Feature selection has a tremendous impact on the performance of the learning algorithm. The selection can lead to better performance and well customization for a targeted task. However, in some cases, the improvement in performance does not necessarily indicate better-generalized performance. If this was the case, the implementation of such is called the overfitting model which performs well in a specific task but performs poorly on other similar tasks. Function words might create extraneous information because the common use of such words by most users decreases the value as markers for authors' styles. However, research shows that function words can be used when coupled with another feature such as punctuation (Jurafsky & Martin, 2009). TF-IDF is commonly used to find similarities based on the analysis of the frequency of terms used across the documents written by the same author (Jurafsky & Martin, 2009). Using TF-IDF, the unique words used by a specific author can be determined, especially after eliminating function words. The n-grams technique studies the sequence of terms that come together through the whole text. It gives a good glance at the probability of a word (e) given word (h); $p(e|h)$.

The use of n-grams has an advantage, as there is no need for linguistic knowledge (Rocha et al., 2017). Another technique is the use of cross-topic or cross-genre attribution. Even though it is not applicable for all scenarios because of the similarity of the topics in a dataset, it can be used in the case of the variant topics or genres involved. As Natural Language Processing (NLP) is becoming more sophisticated, it has paved the way for the inclusion of complicated stylometric features (Argamon et al., 2007), (Gamon, 2004), (Hedegaard & Simonsen, 2011), (Hirst, 2007), (Posadas-Durán et al., 2015), (Stamatatos et al., 2000), (van Halteren, 2004). However, the use of such features is heavily dependent on the NLP toolkit and produces undesirable noise during the training of AA models (Rocha et al., 2017). Besides, POS tagging is another feature that can be used as a marker for authors' writing styles. Complexity analysis of authorship signal is an alternative for the machine learning and stylometric features. It takes the whole text to quantify the signal of the text to be compared later with other signals to determine the similarity between those signals. The use of compression algorithm is a very well-known technique to achieve complexity analysis (Oliveira et al., 2013).

Most teams applied common techniques for feature selection, such as the stylistic feature of the text, including punctuation marks, capital letters, and quotations. A group of teams also selected content features such as Latent Semantic analysis, a bag-of-words, TF-IDF, dictionary-based words, topic-based words, and entropy-based words (Sapkota & Solorio, 2013), (Patra et al., 2013), (Lim et al., n.d.), (Flekova & Gurevych, 2013), (Meina et al., 2013), (Cruz et al., 2013), (Santosh et al., 2013), (Pavan et al., 2013), (Díaz & Hidalgo, 2013). Moreover, emotion words were also considered by (Meina et al., 2013), (Flekova & Gurevych, 2013), (Díaz & Hidalgo, 2013). An N-grams feature was included

by (Meina et al., 2013), (Jankowska et al., n.d.), (Moreau & Vogel, 2013), (Sapkota & Solorio, 2013).

A distinctive feature that was only used by Meina et al. is collocations (Meina et al., 2013). López-monroy et al. was the only team that used the Second-order representation relationship between documents and profiles and achieved the highest accuracy in the age classification task and ranked second in the overall score (López-monroy et al., 2013). Another feature used by Modaresi et al. and Bougiatiotis & Krithara, 2016) is stylistic features with n-gram (Modaresi et al., 2016) (Bougiatiotis & Krithara, 2016). Pimas et al., who achieved the lowest joint accuracy in the task, also used stylistic features (unique words, sentiment words, etc.) (Pimas et al., 2016). However, they used different tree-based algorithms (e.g., Random Forest, J48, LADTree) as classifiers, which we believe led to poor accuracy. Argamon et al. revisited the same corpus; this time, their aim was to study the writing style differences between male and female authors (Argamon et al., 2003). They proposed studying the most distinguishing features for male authors and female authors. They found that determiners, quantifiers, and some POS tags are used by male authors more than female authors. Conversely, female authors use pronouns (I, you, she, her, their, myself, yourself, herself) more than male authors.

Unsolved Challenges

As highlighted in our survey of authorship profiling, many researchers have attempted to solve the problem empirically. The techniques and approaches that have been tried are huge. Based on our observation, the features selection and classification models play an important role in the results of the proposed method. In previous studies of authorship profiling, different features are found to be directly related to the performance

of proposed models. The extensive recent research has shown that some models perform better than others in the task of age and gender classification. However, the attempts have not addressed some of the challenges and have not offered optimal solutions for some unsolved problems.

The area of authorship profiling has been proven to be an important and complicated problem that requires further investigation. To date, there has been little agreement about the nature of the problem we are trying to solve. There is no commonly agreed method for age and gender classification despite the genre or nature of the analyzed text. In addition, no research has been found that surveyed the different approaches that have been used in authorship profiling problems and contrasted between them. What is known about authorship profiling is heavily based on small-scale experiments of the many attempts to tackle the problem of finding a reliable classification method with a near-zero error rate.

Some techniques in NLP have emerged and proved to be state-of-the-art for many NLP tasks (Vaswani et al., 2017). Transfer learning is one of the techniques that has not been fully tested in the context of authorship profiling. A systematic understanding of how transfer learning contributes to authorship profiling is still lacking. Surprisingly, the effects of the emerging of transfer learning on authorship profiling have not been closely examined. Therefore, the need to apply such an advanced and reliable technique as a transfer learning approach to authorship profiling is worth investigating. While transfer learning is a growing field, publications on authorship profiling using transfer learning remain few. Relatively little research has been carried out on authorship profiling and even less on authorship profiling using transfer learning. Despite the importance of transfer

learning in NLP and text analysis overall, there remains a paucity of evidence on the effect of transfer learning on authorship profiling.

Table 1. A summary of literature review (This is a short version, to see the full version see appendix A)

Paper	Accuracy	Preprocessing	Features	Classifier
(Santosh et al., 2013)	0.3508	N/A	<ul style="list-style-type: none"> • Stylistic features (punctuation marks, capital letters, quotations, etc.) • POS tags • HTML, image URLs • Readability • Content features (Latent Semantic Analysis, bag of words, TF-IDF, dictionary-based words, topic-based words, entropy-based words, etc.) 	<ul style="list-style-type: none"> • SVM • Max-Entropy • Decision tree
(Lim et al., n.d.)	0.3488	N/A	<ul style="list-style-type: none"> • Stylistic features (punctuation marks, capital letters, quotations, etc.) • POS tags • Readability • Content features (Latent Semantic Analysis, bag of words, TF-IDF, dictionary-based words, topic-based words, entropy-based words, etc.) 	<ul style="list-style-type: none"> • SVM
(Cruz et al., 2013)	0.3114	N/A	<ul style="list-style-type: none"> • POS tags • Content features (Latent Semantic Analysis, bag of words, TF-IDF, dictionary-based words, topic-based words, entropy-based words, etc.) 	<ul style="list-style-type: none"> • Decision trees • SVM
(Sapkota & Solorio, 2013)	0.2471	N/A	<ul style="list-style-type: none"> • HTML, image URLs • Content features (Latent Semantic Analysis, bag of words, TF-IDF, dictionary-based words, topic-based words, entropy-based words, etc.) • n-grams 	<ul style="list-style-type: none"> • SVM

(Moreau & Vogel, 2013)	0.2395	N/A	<ul style="list-style-type: none"> • n-grams 	<ul style="list-style-type: none"> • SVM • Decision trees • Logistic regression • Naïve Bayes
(Marquardt et al., n.d.)	0.152375	<ul style="list-style-type: none"> • HTML and XML cleaning • Deleted spam bots 	<ul style="list-style-type: none"> • Number of posts per user, the frequency of capital letters, capital words • HTML tags (img,href,br) • Automated readability index • Coleman-Liau Index • Rix Readability Index • Gunning Fox Index • The occurrence of emotions • MRC, LIWC features for psycholinguistic words frequency (familiarity, concreteness, imagery, motion, emotion, religion, etc.) • Identify lexical errors • Specific gender phrase • Sentence sentiment 	<ul style="list-style-type: none"> • SVM
(Álvarez-Carmona et al., 2015)	0.7906	N/A	<ul style="list-style-type: none"> • Latent Semantic Analysis with Second-Order features combination based on the relationship among terms, documents, profiles, and sub-profiles. 	<ul style="list-style-type: none"> • SVM(Lib Linear)
(González-gallardo et al., 2015)	0.774	<ul style="list-style-type: none"> • Hashtags, URLs, mentions 	<ul style="list-style-type: none"> • Character n-grams • POS n-grams • 	<ul style="list-style-type: none"> • SVM(Lib Linear)

(Grivas et al., 2015)	0.7487	<ul style="list-style-type: none"> • HTML code removal • Hashtags, URLs, mentions 	<ul style="list-style-type: none"> • Word length • Letter case • Twitter-specific features (links, hashtags, mentions) • TF-IDF n-grams • 	<ul style="list-style-type: none"> • SVM (age, gender) • Regression (personality)
(Kiprov et al., 2015)	0.7211	N/A	<ul style="list-style-type: none"> • Character flooding • Letter case • Twitter-specific features (links, hashtags, mentions) • A combination of n-gram model • Polarity words, emotions • NRC 	<ul style="list-style-type: none"> • SVM (age, gender) • regression (personality)
(Poulston et al., 2015)	0.6743	<ul style="list-style-type: none"> • Retweets removal 	<ul style="list-style-type: none"> • A combination of n-gram model • Topic modeling with LDA • 	<ul style="list-style-type: none"> • SVM (age, gender), • Regression (personality)
(Bartoli et al., 2015)	0.6557	<ul style="list-style-type: none"> • Retweets removal 	<ul style="list-style-type: none"> • LIWC • Informative words 	<ul style="list-style-type: none"> • SVM (classification), • Random Forest (regression)
(Najib et al., 2015)	0.613	<ul style="list-style-type: none"> • HTML code removal 	<ul style="list-style-type: none"> • Word n-grams • Most discriminant words among classes 	<ul style="list-style-type: none"> • SVM (Lib Linear)

(Nowson et al., 2015)	0.6039	<ul style="list-style-type: none"> • Hashtags, URLs, mentions • Emojis characters removal 	<ul style="list-style-type: none"> • Character flooding • Twitter-specific features (links, hashtags, mentions) • Word n-grams • Named entities recognition • Polarity words, emotions 	<ul style="list-style-type: none"> • SVM (age, gender), • Regression (personality)
(Posadas-Durán et al., 2015)	0.589	<ul style="list-style-type: none"> • Tweets with fewer than 5 words removal 	<ul style="list-style-type: none"> • Twitter-specific features (links, hashtags, mentions) • Ten types of 10 different kinds of n-grams (lemmas, words, relations, POS, etc.) 	<ul style="list-style-type: none"> • SVM(Lib Linear)
(Bougiatiotis & Krithara, 2016)	0.3974	<ul style="list-style-type: none"> • Lemmatization • Lowercase letters • Numbers removal • Hashtags, RTs, URLs 	<ul style="list-style-type: none"> • Stylistic features (unique words, sentiment words, etc.) • Stylistic features with n-grams • Second-order representation 	<ul style="list-style-type: none"> • SVM
(Busger et al., 2016)	0.3846	N/A	<ul style="list-style-type: none"> • Stylistic features (unique words, sentiment words, etc.) • Stylistic with POS • Second order representation 	<ul style="list-style-type: none"> • SVM
(Markov et al., 2016)	0.2949	<ul style="list-style-type: none"> • Numbers removal • Hashtags, RTs, URLs • Transition point techniques 	<ul style="list-style-type: none"> • Stylistic features with n-grams • Second-order representation • 	<ul style="list-style-type: none"> • SVM

(Dichiu & Rancea, 2016)	0.2692	N/A	<ul style="list-style-type: none"> • Verbosity • n-grams weighted with TF-IDF • 	<ul style="list-style-type: none"> • SVM
(Gencheva et al., 2016)	0.2564	<ul style="list-style-type: none"> • Hashtags, RTs, URLs • Feature selections 	<ul style="list-style-type: none"> • Stylistic features (unique words, sentiment words, etc.) • Stylistic features with n-grams • Stylistic with POS • Readability indexes 	<ul style="list-style-type: none"> • SVM with bootstrapping
(Bayot & Gonçalves, 2016)	0.2179	N/A	<ul style="list-style-type: none"> • Word2vec • 	<ul style="list-style-type: none"> • SVM
(Bakkar Deyab et al., 2016)	0.2051	<ul style="list-style-type: none"> • Punctuation signs, stop words removal • Stemming 	<ul style="list-style-type: none"> • Bag-of-words 	<ul style="list-style-type: none"> • SVM
(Basile et al., 2017)	0.7429	N/A	<ul style="list-style-type: none"> • Combination of character 3-5-grams and TF-IDF word 1-2-grams 	<ul style="list-style-type: none"> • SVM
(Tellez et al., 2017)	0.7267	N/A	<ul style="list-style-type: none"> • Bag-of-words 	<ul style="list-style-type: none"> • SVM
(Markov et al., 2017)	0.7125	<ul style="list-style-type: none"> • URLs, mentions, hashtags removal 	<ul style="list-style-type: none"> • n-gram: • 3-4-gram for typed character; 3-7-gram for untyped character; 2-3-gram for words • Domain names 	<ul style="list-style-type: none"> • SVM
(Pastor López-Monroy et al., 2017)	0.7029	N/A	<ul style="list-style-type: none"> • Second order representation based on document relationship 	<ul style="list-style-type: none"> • SVM

(Ciobanu et al., 2017)	0.5904	N/A	<ul style="list-style-type: none"> • Word and character n-gram 	<ul style="list-style-type: none"> • SVM • Ensemble d different algorithms •
(Kheng et al., 2017)	0.5704	<ul style="list-style-type: none"> • URLs, mentions, hashtags removal • Lowercase words • Stop words removal • Short tweets removal 	<ul style="list-style-type: none"> • Word n-gram • LSA 	<ul style="list-style-type: none"> • SVM • Naïve Bayes (gender)
(Ganesh et al., 2017)	0.4713	<ul style="list-style-type: none"> • N/A 	<ul style="list-style-type: none"> • Word with frequency between 2-25 	<ul style="list-style-type: none"> • SVM

CHAPTER 3

METHODOLOGY

In this section, we are explaining the proposed method of our study. The study aims to investigate the feasibility of using PTMs to improve the overall accuracy of profiling the age and gender of anonymous authors. It provides a comprehensive analysis of the associated factors when PTMs are applied toward authorship profiling. To apply PTMs on profiling the age and gender we need to experiment with different PTMs and different hyperparameters. As PTMs are considered features-independent, we need to test the different techniques of text preprocessing. Subsequently, the decision of whether to apply the different preprocessing techniques or not is directly connected to the performance of the resulted model. Based on the literature review, we identified the most used preprocessing techniques. As a result, one of our aims is to examine the impact of the most used preprocessing techniques in profiling age and gender of the author when pretrained models are applied e.g., BERT, RoBERTa, and BERTweet.

Our research also offers a comparative study in which the testing of different PTMs in solving authors' gender and age profiling is examined. We reviewed the approaches that have been proposed to solve the problem of forensic authorship age and gender profiling. To the best of our knowledge, no one has conducted comprehensive experiments testing the impact of using transfer learning models in the domain of forensics authorship

profiling. As transfer learning models are proven to outperform most state-of-the-art models (Devlin et al., 2019), our aim is to test and measure their performances when they are employed to solve the forensic authorship profiling challenges. Challenges that have been mentioned earlier such as, including the lack of scaling with a larger dataset, bias toward the selected dataset, and obstacles posed by the brevity of text in today's communications. We examined three of the most widely used PTMs, and we conducted different experiments to observe how they perform when applied to classify author's age and gender. To answer our research questions and accomplish our contributions, we are considering the following steps as explained in figure 6:

- 1- Choosing more than one PTMs that are known to perform well for classification problems.
- 2- Choosing a dataset for age and gender profiling that has been tried by researchers which has published results and methods that we can compare our method with. In the dataset handling phase, we employed our proposed method of translating the emojis to meaningful words. Further, we added missing tokens by replacing missing tokens with existing predefined ones to boost tokenizer's text representation. We also compared the analysis of individual tweet vs. the combined long tweets in the case study of profiling fake news spreaders. Finally, we experimentally examined the best text preprocessing techniques for profiling the age and gender of authors.
- 3- Implementing custom software that can accommodate our mentioned goals in points 1 and 2. The software will have a generic implementation and trained and tested on the selected dataset to be introduced as a general implementation.

- 4- Generalizing our approach as much as we can to help to standardize forensic authorship profiling using the transfer learning technique.
- 5- Testing our model against a new dataset and utilizing the extra layer of confidence to optimize the model performance.
- 6- Choosing common metrics such as the F1 score to easily compare the results of selected pretrained models.

The main goal of our research is to test the viability of using PTMs for forensic authorship profiling. In the meantime, we thoroughly analyzed all the associated factors

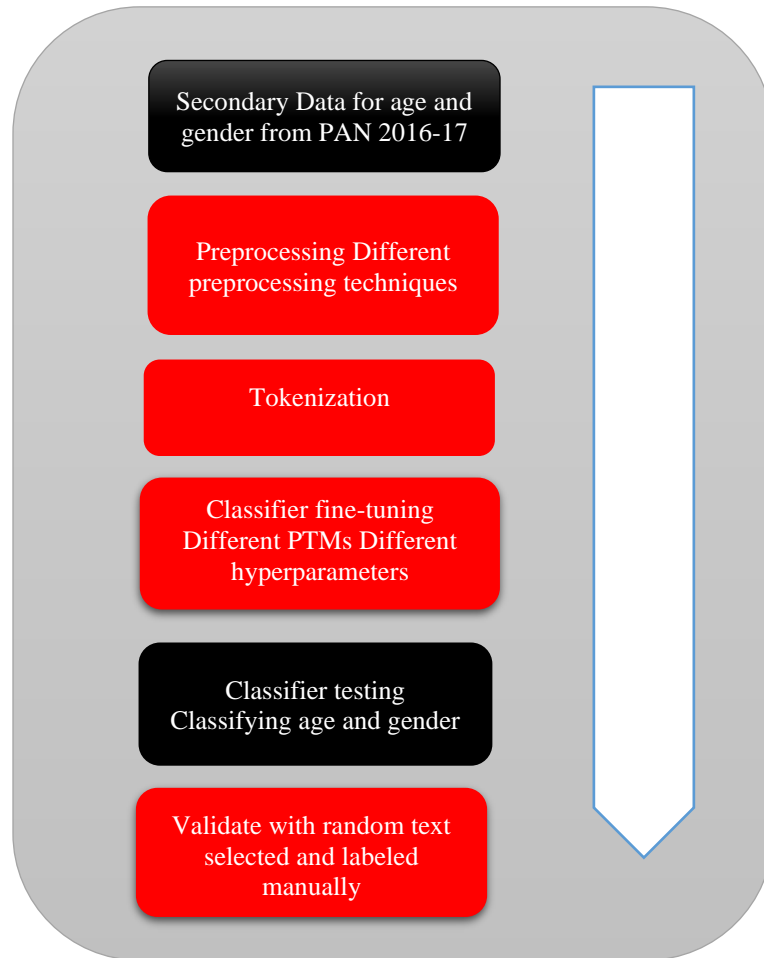


Figure 6. General implementation pipeline

of applying PTMs in the domain of forensic authorship profiling. We also explored how different PTMs perform when applied toward profiling the ages and genders of authors. Moreover, we learned more about the limitations and challenges of using such a technique in the context of authorship profiling to offer a comprehensive analysis that can be used as a guideline for future research.

Traditional Vs Transfer Learning

The implementations of traditional and transfer are almost similar, but they differ in the pretraining and the downstream parts. Figure 7 illustrates the phases of transfer learning. As the name suggests, in transfer learning, we transfer the knowledge of a model that was trained on a different dataset that is closely like the domain of the downstream task. The dataset in the pre-training phase usually is larger compared to the downstream's dataset. Hence, it helps to overcome the limitation of overfitting or small dataset in previously common methods. Instead of training the target or downstream task from scratch, PTMs allow us to transfer the knowledge learned in the pre-training phase to achieve a better result. The pretraining phase usually follows an unsupervised learning technique in which the model learns from an unlabeled dataset (Q. Yang et al., 2020). Fine-tuning is the phase in which we tune the pretrained model to fit the target task. In most cases, downstream follows a supervised technique. Because of the mixed learning methods, supervised and unsupervised, that are used to achieve the pretraining and downstream, the learning method of transfer learning is known as semi-supervised learning.

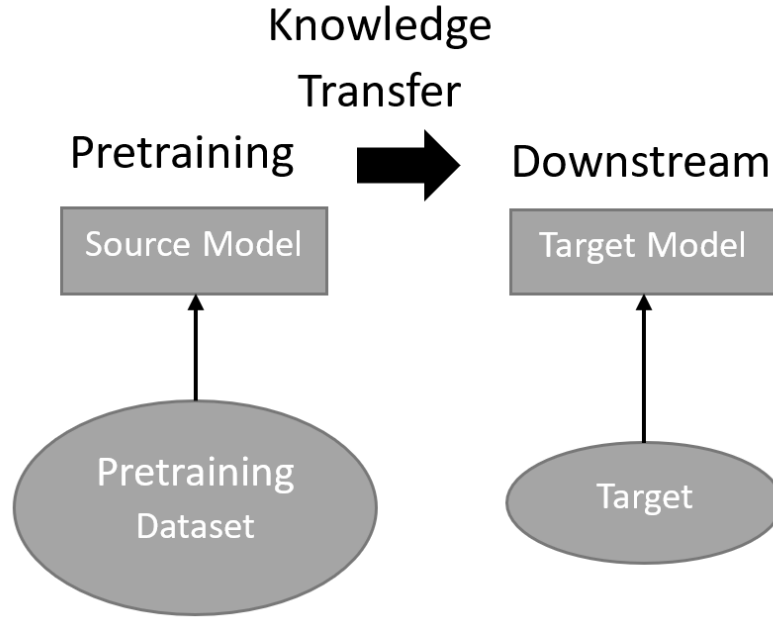


Figure 7. The process knowledge transfer

Dataset

The dataset we used is taken from author profiling PAN shared (Rangel et al., 2016). First dataset was used in 2016 Authors profiling shared task by PAN. We were granted access to the data by the task organizers on Aug 8th, 2020. We chose to work on the training dataset only because it is large enough. Moreover, the evaluation of the testing dataset is blinded and can only be done by the PAN committee. Also, we only chose to conduct our experiments on the English corpus due to the focus and scope of our study. The selected dataset has two labels, author's gender, and age. The age is categorized as follows: 1) 18-24; 2) 25-34; 3) 35-49; 4) 50-64; 5) 65 and above. Table 2 shows the distribution of the dataset per class.

Table 2. The distribution of dataset per class

Number of authors	436
Number of tweets	363,031
Number of tweets per male authors	149059
Number of tweets per female authors	113972
8-24	18126
25-34	92059
35-49	105520
50-64	44896
65 and above	2430

The second dataset we used is the dataset of author profiling task in 2017 (Rangel et al., 2017). The dataset consists of different languages text i.e., English, Arabic, Spanish, and Portuguese. Because our focus is on English text, we excluded all non-English sub-datasets. The literature shows that most studies in the domain of authorship profiling consider datasets that are collected from Twitter. The features of today's writing, such as unstructured, brief, and colloquial, are reflected in Twitter texts. The text of the tweets, along with a truth table including the authors' gender and origin, were provided in the PAN shared task in 2017. The following origin categories were assigned: 1) American, 2)

Australian, 3) Canadian, 4) British. The dataset was also labelled for gender i.e., women and men.

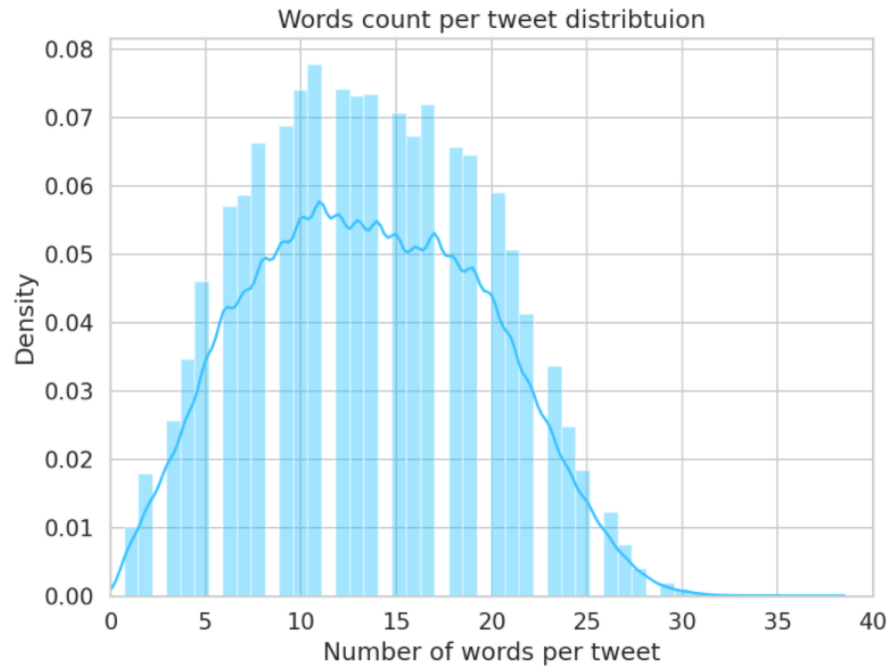


Figure 8. Words count per tweet for the dataset from PAN 2016

Figure 8 shows the distribution of the number of words per tweet throughout the dataset despite the author's classification. Tweets that have between 6-22 words have a higher frequency than others. They always represent at least 5% of the dataset. At 8% of the tweets have less than 5 words which might create ambiguity to the target model. There are ways to avoid such under-representation such as text augmentation or text combination. In some cases, researchers remove short texts that have less than five words.

Figure 9 shows the distribution of the number of words per tweet. As observed, tweets that have between 5-23 words have a higher frequency than others. Hence, they always represent at least 5% of the dataset. Another important observation is that at 7% of

the tweets have less than 5 words which might create ambiguity to the target model. In the effort of normalizing the dataset, different techniques need to be considered to better represent the dataset and ensure the quality of the dataset before feeding it to the target model.

Table 3. The dataset statistics (The distribution of tweets per gender and the total number of authors)

Number of authors	3600
Number of tweets	360,000
Number of tweets per men	180,000
Number of tweets per women	180,000

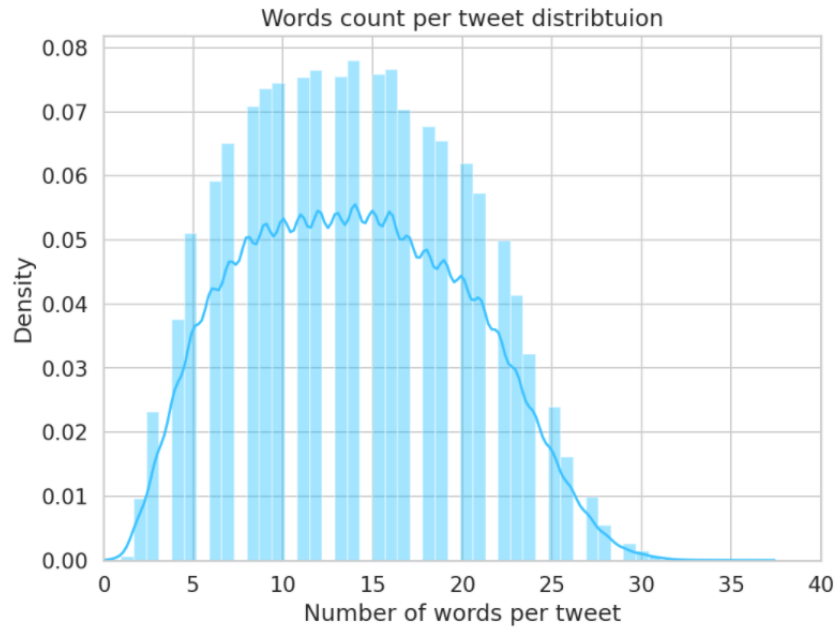












Figure 9. Words count per tweet for the dataset from PAN 2017

The Dataset from PAN 2017 author profiling differs from the one from 2016 because in 2017 the dataset has emojis involved. Since OSNs are known for informal communication, the use of emojis in such platforms is considerable. As a result, we considered studying a dataset that has emojis to examine the effect of different emojis-related text-preprocessing techniques. Furthermore, there are Python libraries that are designated to handle the existence of emojis within a text. Table 4 show the 10 most frequent emojis in the dataset and emphasize the heavy use of emojis in the dataset.

Table 4. The most 10 frequent emojis in the dataset from PAN 2107

Emoji	Frequency
	11172
	3759
	2907
	2617
	2240
	2237
	1961
	1716
	1655
	1295

Experiments

Preprocessing techniques vary depending on the nature of the text and the objective of the task. In the past, text analysis used to concern formal long texts such as books, poems, literature, and plays. Currently, the focus of text analysis has shifted towards the short informal unstructured texts such as OSNs' textual content. Preprocessing in traditional techniques greatly matter, but in transfer learning, PTMs need every part of the target text to ensure the successful convergence of PTMs in downstream. The objective of the task plays an important role in the direction of the implementation of the downstream. PTMs have been extensively tested on different NLP prediction tasks such as the General Language Understanding Evaluation (GLUE), the Stanford Question Answering Dataset (SQuAD v1.1), and the Situations With Adversarial Generations (SWAG) (Devlin et al., 2019). PTMs for classification have been tried for the tasks of sentiment analysis. In our study, we approach the problem as a classification task on which we classified the text based on the author's gender and age. Then, we profiled the authors of texts based on the classification of their published texts.

In transfer learning, the type of tokenizers is directly linked to the performance of the implemented models. PTMs are mostly non-features-dependent models in which the wide features extraction and selection are not common with the context of transfer learning. Hence, PTMs are more stable and resistant to the big variations that are common in traditional learning. In some tasks, researchers have used the same models on the same data, and yet they achieved significantly distant different results. This happened due to the dependency of such models on features extraction and selection. PTMs overcome that by focusing on language modeling architecture-based in which text tokenization is the most

relevant step before training PTMs. PTMs tokenizers vary from model to model e.g., Byte-Pair Encoding (BPE), Byte-level BPE, WordPiece, unigram (used in conjunction with SentencePiece), and SentencePiece.

The pre-trained models we can use in our study:

- Bidirectional Encoder Representation from Transformer (BERT)
- Optimized BERT Pre-training Approach (RoBERTa)
- Pre-trained language model for English Tweet (BERTweet)

BERT	RoBERTa	BERTweet
<ul style="list-style-type: none"> • Bidirectional • Transformer Encoder • Pretrained on WikiEn and BookCorpus • Achieved SOTA in 11 NLP tasks (the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answer- ing Test F1 to 93.2 (1.5 point absolute im- provement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement). 	<ul style="list-style-type: none"> • Based on BERT • Pretrained on CC-NEWS dataset • Establishes a new state-of-the-art on 4/9 of the GLUE tasks: MNLI, QNLI, RTE and STS-B • Uses different tokenizer, larger byte-level BPE vocabulary 	<ul style="list-style-type: none"> • Based on BERT • Pretrained on large Twitter dataset collected from xxxx-xxxx • Custom for Twitter dataset tasks • Consideration of emojis and emoticons • Outperforms some SOTA models in 3 Twitter NLP tasks: Part-of-speech tagging, Named-entity recognition and text classification.

Figure 10. Examples of PTMs

Figure 10 shows few examples of PTMs and their differences in terms of implementation and pretraining datasets. Most PTMs are based on the transformer where some PTMs utilize the transformer’s decoder and others utilize the transformer’s encoder. The PTMs also vary in the type of tokenization, the number of tokens and the techniques of language modeling they use (unidirectional and bidirectional.)

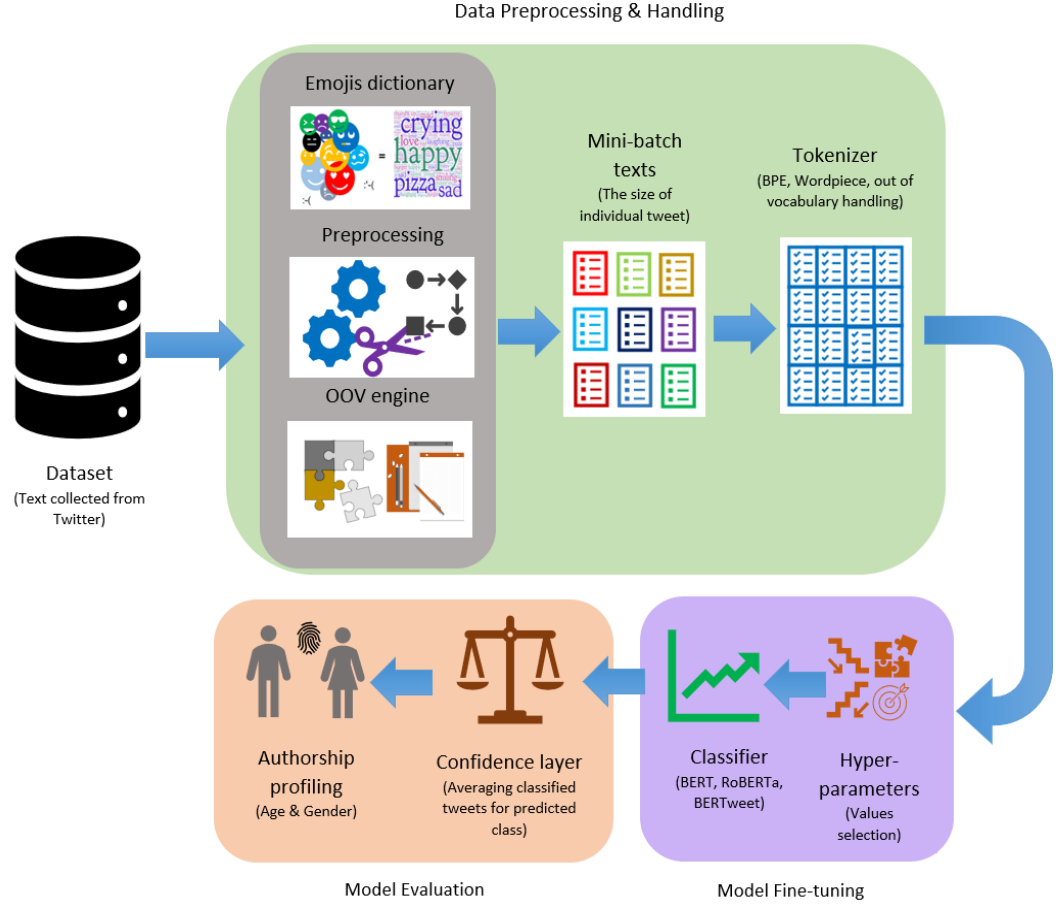


Figure 11. Examples of PTMs (The design of our proposed approach)

As mentioned before, we considered using transfer learning techniques for this task. Transfer learning has been proven to be state-of-the-art for many tasks (Devlin et al., 2019). There are different choices we could have considered, but in the interest of the scope of this study, we have considered using BERT, RoBERTa, and BERTweet using Huggingface library. With the use of Huggingface, we can run multiple experiments with different parameters for better accuracy. After conducting multiple experiments, we were able to observe the best parameters values and techniques that are associated with the implementation of BERT, RoBERTa and BERTweet. All the experiments were carried out using Google Colab’s graphical processing unit (GPU) to optimize the time efficiency. The

different chunks of the code serve the purpose of each aim we are proposing, Preprocessing techniques effect, different tokenizers effect, and different PTMs comparison.

Dataset: The dataset we used is from PAN’s 2016 shared tasks involving author profiling (Rangel et al., 2016). We chose to conduct our experiments on the English corpus only due to the focus and scope of our study. The most studied datasets in the literature are collected from Twitter. Twitter texts represent the characteristics of today’s text e.g., unstructured, short, and colloquial

Emojis dictionary: The dictionary contains the interpretation of emojis and emoticons in meaningful words. BERT tokenizer does not recognize emojis and emoticons. Instead of assigning a unique code as it does with words, BERT assigns unknown (UKN) tokens to emojis and emotions. We programmatically added corresponding tokens for all emojis and emotions to be recognized by BERT. Besides, there is a custom version of BERT called BERTweet which was built with consideration of pretraining on the large corpus from Twitter and the implementation of a custom tokenizer with consideration of emojis, emoticons, Twitter-specific vocabulary. This solved the problem of out-of-vocabulary which limits the performance of models by adding many unknown tokens (UKN).

Preprocessing engine: There are different text-preprocessing techniques in NLP. In our study, we are examining the most common techniques that have been applied by researchers to see the effect of such techniques when BERT model is used to profile age and genders.

Out-of-Vocabulary engine: Out of Vocab is a serious issue in NLP. Even though the implementation of models have been advanced, we still see some challenges at the level

of text representation. As a result, tokenization of text will be affected which eventually will affect the model performance because of the missing tokens. When we have missing tokens the model tokenizer cannot encode and decode the text efficiently. So, in this engine we experimented different techniques hoping to mitigate the effect of this problem.

Mini-batch texts: In this approach, we divided each author's text into tweet size batches and feed them to the tokenizer. This scenario allowed us to add an extra layer for confidence calculation. As the collected dataset is divided into tweets, we processed each author's tweets individually and ensure including all tweets for the same author in the same set of split datasets e.g., training, testing, and evaluating datasets.

Tokenizers: BERT uses WordPiece tokenizer and RoBERTa uses Byte-Pair Encoding tokenizer. Details and examples are provided below. Our implementation considered analyzing and mitigating the effect of out of vocabulary problem.

Hyper-parameters values selection: Each model has different hyper-parameters that can have different values. The developers of each model usually recommend a range of certain values for each hyper-parameter to achieve similar results that are achieved by the model's developers. For example, the authors of BERT (Devlin et al., 2019) suggested the number of epochs, batch size, and learning rate.

Classifier: In our study, we used BERT, RoBERTa, and BERTweet pretrained models utilizing the Huggingface transformer library and Google Colab virtual environment.

Confidence layer: It calculates the percentage of classified tweets by the model and determines the author's gender and age. If the determined class is above the set threshold, the class was determined accordingly. The suggested threshold was determined after

experimenting with different hyper-parameters values and the resulted accuracy of the classifier before considering the threshold of the confidence layer.

Age and gender profiling: The trained models were tested to profile the age and gender of authors utilizing the proposed techniques of the dictionary of emojis and emotions and the confidence layer

Profiling Hate Speech Spreaders By Classifying Micro Texts Using Bert Model (Case Study)(Esam & Leon, 2021)

As the first step in most text analysis tasks, preprocessing is an important step that can significantly affect model performance. In our method of handling the datasets, we have tried different techniques with controlling the method of classification to ensure the best use of the given datasets. We have found that the English dataset gives a better performance when hashtags, URLs, mentions, RT tags, and punctuation are removed. However, the Spanish dataset gave better accuracy when we kept the text unchanged.

Because we used the transfer learning method for our classification, we have conducted experiments about the tokenization of different transfer learning models such as BERT, ROBERTA, and Longformer. BERT uses WordPiece tokenizer¹; one of the downsides of WordPiece is the lack of emojis support. Emojis are heavily used in social media contexts, and they can reveal useful information that can boost model performance. On the other hand, RoBERTa and Longformer use the same tokenizer, Byte-Pair Encoding (BPE). BPE can handle emojis and assign a token to them. Conversely, WordPiece assigns an unknown token [UNK].

As mentioned before, we considered using transfer learning techniques for this task (Esam & Leon, 2021). Transfer learning has been proven to be state-of-the-art for many

tasks (Devlin et al., 2019). There are different choices we could have considered, but in the interest of time and simplicity, we have considered using BERT, ROBERTA, and Longformer Huggingface implementations. With the use of Huggingface, we conducted experiments with different parameters for better accuracy. After conducting multiple experiments, we found that BERT gave us the best accuracy for the target task. For the English dataset task, we used bert-base-cased version with the following hyperparameters.

- Batch_size = 32
- Epochs= 50
- Learning rate = 2-5
- Max_length = 50

For the Spanish dataset task, we used bert-multilingual-cased with the following hyperparameters.

- Batch_size = 16
- Epochs= 50
- Learning rate = 2-5
- Max_length = 60

The hyperparameters are within the range of suggested values by BERT authors (Devlin et al., 2019).

After visualizing and understanding the target datasets, we decided to handle the data differently and observe the outcome. We built three scenarios: 1) using the preprocessed dataset as is; 2) combining all tweets for each user; and 3) considering all tweets for each user to be in the same group after splitting the dataset into training,

validation, and testing datasets (Fig2 shows the steps). The first scenario gave the best accuracy among all three scenarios. However, we could not test the accuracy for each user because the tweets were randomly selected and not all users have the same number of tweets. The second choice was the worst among all the scenarios as the dataset became small, which limited the ability of the model convergence and representation.

The results are based on the first scenario for both the Spanish language and the English language. The third scenario gave us the ability to test the model within 10 balanced authors, including all the tweets for all selected authors, Figure 12 illustrates the third scenario steps where all tweets for the same user are included in the assigned dataset (Training, testing, validation) with a random shuffle. This scenario allows us to add an extra layer for confidence, which calculates the number of tweets that are classified as hateful for each user. In our initial testing, we achieved 80% accuracy when we determined the threshold for the number of hateful tweets for each use at 95. If more than 95 tweets of any user were classified hateful, the user will be classified as a hate speech spreader.

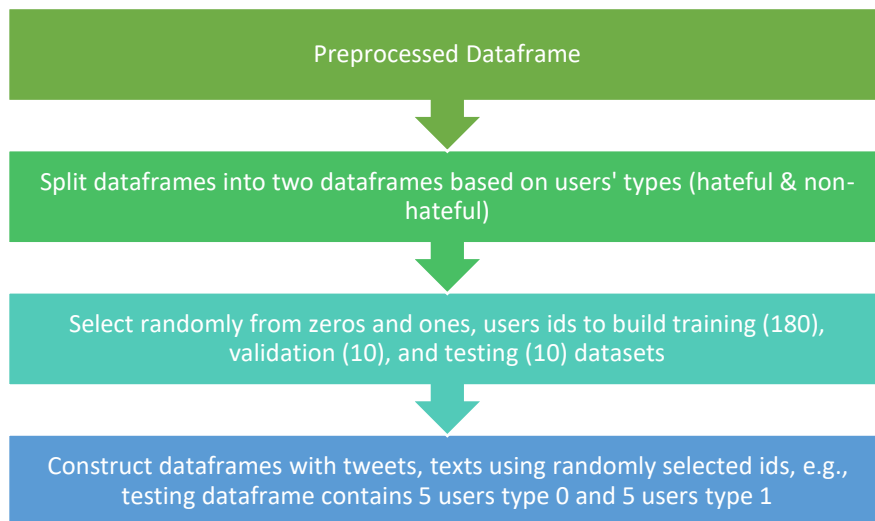


Figure 12. The 3rd scenario dataset splitting

Preprocessing Techniques Effect (Alzahrani & Jololian, 2021)

Because transfer learning does not require much consideration of features engineering, the only obvious parameter that can vary from one study to another is the preprocessing techniques. Further, the obvious advantage of examining those cases is to determine the best preprocessing technique for author profiling when transfer learning is used.

In terms of applying that programmatically, regex is a powerful way of applying rules that determine what to leave and eliminate from any texts. We built different expressions that handle one part of the text at a time e.g., an expression that removes all hashtags from the processed text. Table 5 shows some examples of the use of regex in tweets preprocessing techniques. Preprocessing techniques have been covered thoroughly in the literature review chapter. Since PTMs need all data to model the language and develop a better understanding of the text, we limited the techniques we were applied.

Hence, we ensure, as much as possible, that our models were independent of any unrelated techniques which are not part of the PTM itself. However, some parts of texts can create some noise which might affect the performance of the model. Our techniques were limited to whether we remove Twitter's specific features like mentions, retweet tags, hashtags, URLs, and media encoding. In our preprocessing experiments, The experimental setups adopted were based on the most common techniques observed in the literature (Rangel et al., 2013b, 2016, 2017, 2015; Rangel & Chugur, 2014), all of which have been extensively tested in the context of author profiling using machine and deep learning techniques. The effect of preprocessing techniques on author profiling using transfer

learning techniques has not yet been studied, However, as illustrated in Table 6, we considered seven cases for the preprocessing techniques.

Table 5. Examples preprocessing using regex in python

Regular expression in Python	Effect
<code>regex.sub(r'(?:@[\w_]+)', u'', tweet)</code>	Mentions removal
<code>regex.sub(r"rt ", u'', tweet)</code> <code>regex.sub(r"RT ", u'', tweet)</code>	Retweet tags removal
<code>regex.sub(r"(?:\#+[\w_]+[\w'\"]*[\w]+)", u'', tweet)</code>	Hashtags removal
<code>regex.sub(r"http\S+", u'', tweet)</code>	URLs removal
<pre> for w in word_tokens: if w not in self.stop_words: w = stem(w) filtered_line.append(w) return ' '.join(filtered_line) </pre>	Stop words removal
<pre> from nltk.stem import WordNetLemmatizer lemmatizer = WordNetLemmatizer() </pre>	Lemmatization
<pre> from nltk.stem import PorterStemmer porter = PorterStemmer() </pre>	Stemming

In Case 1, we included three basic techniques: mentions removal, retweet tags removal, and hashtags removal. In Case 2, we added URLs removal to the techniques from Case 1, and in Case 3, we added the removal of punctuation. In Case 4, we applied a well-known technique in NLP, stop words removal, which involves eliminating extremely common words that are liable to be repeated in many texts and that some researchers

characterize as noise, not as markers. In Case 5, we chose not to apply any preprocessing technique in order to gauge its effect.

Table 6. Preprocessing cases

Case	Preprocessing techniques
Case 1	Mentions removal Retweet tags removal Hashtags removal
Case 2	Mentions removal Retweet tags removal Hashtags removal URLs removal
Case 3	Mentions removal Retweet tags removal Hashtags removal URLs removal Punctuation removal
Case 4	Mentions removal Retweet tags removal Hashtags removal URLs removal Punctuation removal Stop words removal
Case 5	None (i.e., each text as-is)

We chose to run each case’s code for three epochs, as suggested by the BERT and RoBERTa model’s developers (Devlin et al., 2019; Liu et al., 2019). Additionally, we

separated each code for each case and manually double-check the effect of the preprocessing technique performed on the dataset. Finally, we built all five cases using regex and NLTK toolkit in Python and the Huggingface transformer library utilizing Google Colab virtual environment.

The rest of the code concerns the implementation of the BERT, RoBERTa, and BERTweet models for binary classification: 0 for authors who are women, 1 for authors who are men. For the age classification, we have 5 classification categories: 0 for age group 18-24; 1 for age group 25-34; 2 for age group 35-49; 3 for age group 50-64; finally, 4 for age group 65 and above. The only difference in each experiment is in preprocessing; the rest of the experiment parameters were controlled and the same. The models that we used are the uncased BERT base and RoBERTa model with hyperparameters listed in Table 7. For this part of the study, the values of the hyperparameters and the types of the BERT, RoBERTa, and BERTweet models considered were not in focus; therefore, we did not dedicate much time to experimenting with different values of the hyperparameters. Instead, we selected hyperparameters' values based on the comparative study of BERT, RoBERTa, and BERTweet which we conducted as a part of this research.

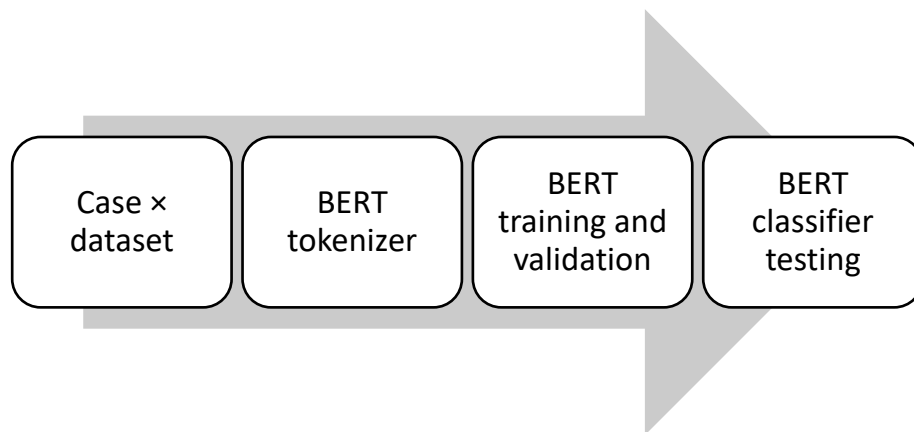


Figure 13. Experiments pipeline for preprocessing techniques study

To better measure the effect of the various preprocessing techniques, we controlled the values of the hyperparameters in all five experiments for each model.

Table 7. Values of the hyperparameters for preprocessing effects

Parameter	Value
Model	BERT-base-cased, RoBERTa, BERTweet
Epochs	3, 4, 5
Batch size	16, 32
Text max length	100
Learning rate	2^{-5} , 5^{-5} , 3^{-5}

In figure 13, the sequence of our experiments' steps and details are demonstrated. There is also a very important factor when fine-tuning PTM for a downstream which is the model's hyperparameters.

The first step in every text analysis task is data pre-processing, or cleaning, in which researchers and/or developers remove any noise that could hinder the model performance. There are multiple pre-processing techniques, including stemming, lemmatization, and lowercase conversion. In a previous study, we examined the effect of pre-processing techniques on profiling the ages of authors, and we concluded that of the pre-trained language models, BERT performed the best when no pre-processing techniques were applied [20].

The dataset must then be tokenized prior to being fed into the model. PTMs are already equipped with tokenizers. For example, as it relates to the three models selected for the present study, BERT uses the WordPiece tokenizer to convert text into meaningful numerical values, and BERTweet utilizes an enhanced version of WordPiece tokenizer, which was trained on a large Twitter dataset from 2012–2020 [17], [21], [22]. RoBERTa employs byte-pair encoding (BPE) to encode the text; because words are represented in BPE by a combination of characters and word-levels, rather than deriving entire words through a statistical analysis of the training corpus, natural language corpora can accommodate immense vocabularies. It should be noted that RoBERTa and BERTweet were both built upon BERT, which was the first model to incorporate transformer implementation. We compared the models using the same hyperparameters, which are delineated in Table 8.

Table 8. Hyperparameters and values

Parameter	Value
Model	bert-base-cased, RoBERTa, BERTweet
Epochs	4
Batch size	8, 16, 32
Maximum text length	100 tokens
Learning rate	2^{-5} , 3^{-5} , 5^{-5}

We utilized the HuggingFace library for PTM on a GoogleColab GPU virtual machine. We separately run each model and perform the same age and gender classification tasks repeatedly. We trained each model for four epochs to ensure convergence; this was a time-consuming process that required more than two-and-a-half hours for each model. The algorithm illustrated in Figure 14 demonstrates the implementation phases to execute model tokenization, training, validation, and testing.

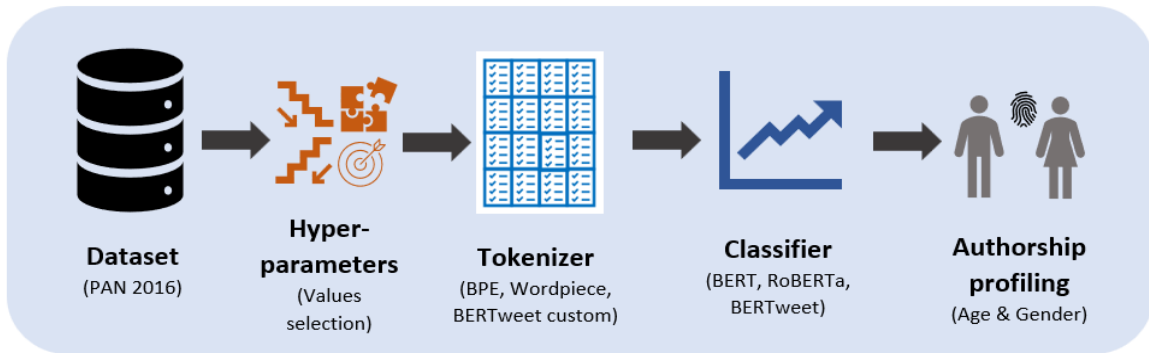


Figure 14. Standard implementation steps to build a pre-trained model that profiles authors' ages and genders.

Three pre-trained models were considered in our study. Each model developer recommended different hyperparameter values, and we selected learning rates and batch sizes in accordance to the recommended values. The suggested learning rate values were 2^{-5} , 3^{-5} , and 5^{-5} ; while there were several suggested batch sizes, in the interests of time and based on the scope of our study, we used 8, 16, and 32. These hyperparameters required a total of 18 experiments for each model to complete the age- and gender-prediction tasks; as a result, we ran 54 experiments to determine the optimal hyperparameter combination for each model to achieve each task. The details of each experiment are discussed in the following section.

Table 9. The suggested values hyperparameters

Batch size	16, 32, 64
Number of epochs	2, 3, 4
Learning rate (Adam)	5e-5, 3e-5, 2e-5

By using three different models, we also discovered how much similar different transfer learning models are and how preprocessing techniques might affect them. In our last experiments, we used different values of the hyperparameters to examine the best values that achieve the highest accuracy. We have three different dependent parameters that can get different values as shown in Table 9. When BERT paper was published (Devlin et al., 2019), the authors recommended some parameters with which we can reproduce similar results. The suggestions were made for three parameters, number of epochs, batch

size, and learning rate. All considered hyperparameters values were tested for BERT, RoBERTa, and BERTweet to profile the age and gender of the author.

The Analysis of Bert Tokenizer

```
[47] 1 txt= df['tweet_text'].iloc[436]
      2 print(txt)
      3 print("BERT tokenizer result:", TOKENIZER.tokenize(txt))
      4 print("ROBERTA tokenizer result:", TOKENIZER2.tokenize(txt))
      5 print("Longformer tokenizer result:", TOKENIZER3.tokenize(txt))

I 'm waiting for that 😊
BERT tokenizer result: ['I', "'", 'm', 'waiting', 'for', 'that', '[UNK]']
ROBERTA tokenizer result: ['I', "Ġ", 'm', 'Ġwaiting', 'Ġfor', 'Ġthat', 'ĠðŁŤĺ', 'Ġ']
Longformer tokenizer result: ['I', "Ġ", 'm', 'Ġwaiting', 'Ġfor', 'Ġthat', 'ĠðŁŤĺ', 'Ġ']
```

Figure 15. The result of text tokenization using different techniques

It is known that emojis are heavily used in social media contexts, and they are characterized by their ability to reveal useful information that can boost model performance. On the other hand, RoBERTa and Longformer use the same tokenizer, BPE (Huggingface). BPE can handle emojis and assign a token to each emoji. Conversely, WordPiece assigns an unknown token [UNK] to all emojis despite the type of the emoji as shown in Figure 15. In part of our study, we examined the effect of using different tokenization techniques, WordPiece and BPE, on profiling the age and the gender of the author. As the type of tokenizer handles text differently, with the use of the same mentioned setup, we observed the effect of BPE and WordPiece tokenizers on profiling the age and gender of the author. Out of Vocabulary is another major challenge in NLP that limits any model from encoding and decoding datasets vocabulary efficiently. If a word or sub-word is not included in the tokenizer of a model, the model will divide the word into sub-characters that are predefined in the model tokenizer. If the symbol is not predefined like emojis and emoticons, the tokenizer will assign an unknown token to the word which leads to the mentioned issue, out of vocabulary. To overcome that, there are some techniques (e.g., text normalization, lemmatization, and stemming) by which the occurrence of

out of vocabulary issues will be reduced. In our study, we implemented a layer that checks for out of vocabulary and update the tokenizer accordingly. We believe that by adding this layer, we can visualize the limitation of BERT tokenizer. Another linguistic technique is called lemmatization. Lemmatization concerns finding the root of each word which helps in normalizing the text. Figure 16 illustrates the different layers a preprocessed text will go through before tokenization.

The noise associated with unstructured text collected for research purposes is overlooked. Researchers usually focus on the type of model and feature engineering. There are noticeable efforts at text representation, e.g., Bag of Words, one hot encoding, embedding, and, more recently, contextualized word representation. Hence, the advancements in such techniques led to a better text representation. However, the cleaning and preparation of text exclusively for target tasks is forgotten. Issues like out-of-vocabulary always appear with text tokenization due to the difference in the target task domain or the rapid change in word use. With recent advancements in transformer-based models like BERT, GPT-3, and others, this issue still hinders the performance of those sophisticated pre-trained models.

The first phase of dataset manipulation is translating emojis and emoticons to meaningful text. As mentioned, the use of emojis in today's communications is immense. Thus, the benefit of translating the emojis to be used by BERT cannot be overlooked. We used the *cPickle* library to translate emojis and emoticons. The *cPickle* library comes with a pre-built emojis-words dictionary.

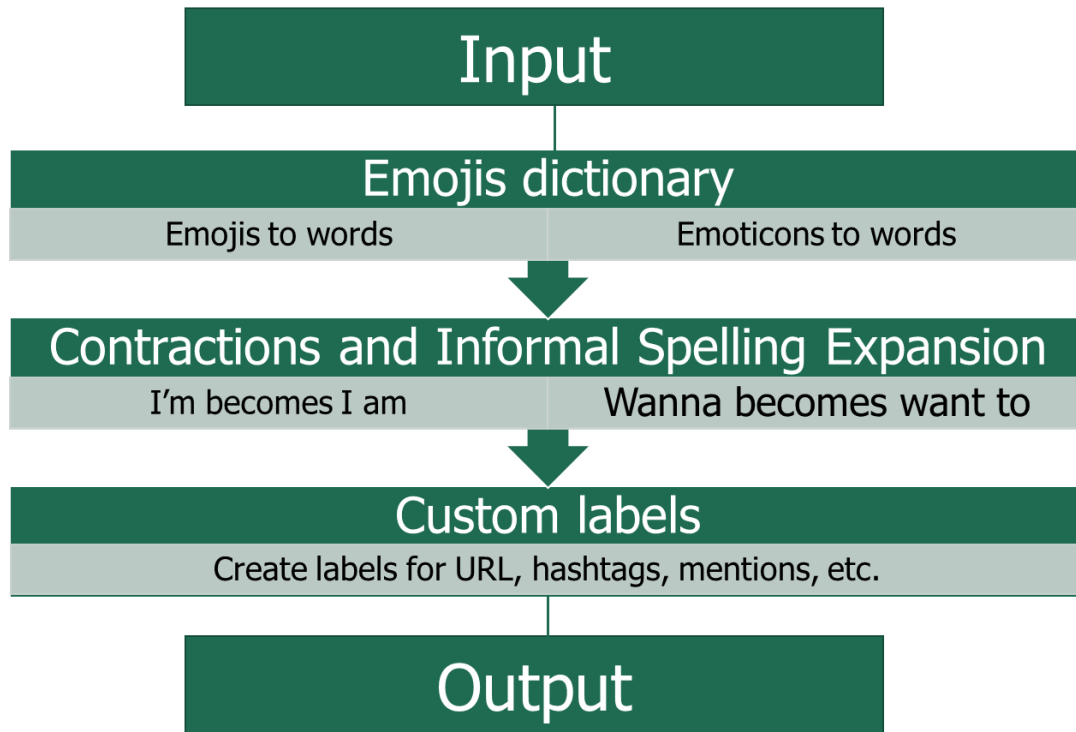


Figure 16. Out of Vocabulary mitigation model

We performed some statistics to check for emojis before we translated them. Table 4 on page 54 shows the top 10 emojis, an extended version of the table showing all the emojis we found in the PAN 2017 dataset is shown on page 130. An example in Table 10 shows the effect of emojis translation on the BERT tokenizer. We can see that the [UNK] token is not present after the emoji translation. We also see how the text was extended, which led to more tokens produced by BERT.

During the analysis of the use of pre-trained models in authorship profiling, we decided to perform text cleaning and noise reduction before the step of tokenization. We believe that by doing so, we will have more meaningful and better-represented text for the

tokenization. Moreover, these steps will help normalize the text and generally reduce the issues tokenizers face. In the text normalization effort, we extended the contractions and

Table 10. The result of tokenization tweets before and after emojis translation

	Tweet text	BERT tokens
Before emojis translation	@NicholasHoult When are Newness, Collide & Rebel in the Rye hitting the theaters?? I want details. Please? 😊	['@', 'nicholas', '##ho', '##ult', 'when', 'are', 'new', '##ness', ',', 'col', '##lide', '&', 'amp', ',', 'rebel', 'in', 'the', 'rye', 'hitting', 'the', 'theaters', '?', '?', 'i', 'want', 'details', ',', 'please', '?', '[UNK]']
After emojis translation	@nicholashoult when are newness, collide & rebel in the rye hitting the theaters?? i want details. please? grinning_face_with_smiling_eyes	['@', 'nicholas', '##ho', '##ult', 'when', 'are', 'new', '##ness', ',', 'col', '##lide', '&', 'amp', ',', 'rebel', 'in', 'the', 'rye', 'hitting', 'the', 'theaters', '?', '?', 'i', 'want', 'details', ',', 'please', '?', 'grinning', ',', 'face', ',', 'with', ',', 'smiling', ',', 'eyes']

presented the full version of the contracted words. The BERT tokenizer does not recognize contractions; instead, it deals with them differently and misrepresents the intended meaning or use. For example, the word “I’m” is tokenized as “[I] [’] [m].” This will prevent BERT from using a proper token that can be used as an important marker for text classification. To perform that, we built a custom dictionary with a good number of contractions. The custom dictionary is shown in Appendix D. Even though there is a ready contractions library, for some reason, the contractions were not fully expanded. However, when we applied the custom contractions expansion we built, all the targeted contractions were fully expanded.

The ability to use the custom-built contractions dictionary is a major advantage of facilitating the process of contractions extension. Hence, we were able to expand all contractions and guarantee the increase of text density. We learned from our text pre-processing study that BERT performs better when more tokens are present; no pre-processing techniques were applied. Thus, by contractions expansion, the BERT tokenizer will have more tokens to represent instead of using sub-words embedding.

BERT uses sub-words embedding to break down words that are unknown to the tokenizer, words that were not encountered by BERT pre-training. For example, any token starts with the symbol `##` to indicate sub-word tokenization. Figure 18 shows the total number of tokens generated from the selected dataset. The single character is the most frequent token in the set, with more than 2.5 million single characters. The other type of token in BERT is sub-word tokens, where the BERT tokenizer chops unknown vocabulary into sub-words to minimize the [UNK] tokens. The percentage of sub-word tokens to the total number of tokens is about 29%. We have rethought the problem by applying contractions extension and spelling correction to minimize the number of sub-word tokens. The behavior of the BERT tokenizer, Wordpiece, is problematic in some cases. In Figure 17, we provide an example of how BERT will deal with “unfollowed,” “followed,” and “un-followed.”

The next step towards text normalization is spelling correction. Informal writing, especially the writing of today’s internet users, suffers from unusual spelling or explicit spelling errors. Again, the BERT tokenizer will struggle to recognize such words and assign them sub-word tokens (`##`). Yet again, the tokenizer and the model will not use the UNKs as markers or information that helps classify texts. To eliminate spelling errors, we

utilized the textblob library. An example of using textblob is the following:
 "His views are very interestng, I like his answr" converted to "His views are very interesting, I like his answers."

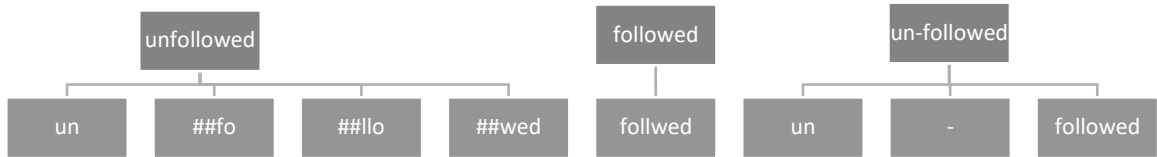


Figure 17. The tokenization of the word “unfollowed” vs. “followed”

The second phase of dataset manipulation is creating custom labels and feeding them to the BERT tokenizer. The labels we consider are custom to Twitter, e.g., URL, mention, hashtag, and media. This will help identify the labels on both text and tokenizer. These labels will be utilized as markers for text classification when associating them with a certain group. BERT will again break down the account name starting with the symbol “@,” the tag that starts with “#,” or the URL that starts with “http.” This will help reduce the noise created by the content of the mentions, URLs, and tags. In the case of age classification, there might be an age group that uses hashtags more than the others. As a result, the model will use that information to classify texts accordingly. An example is shown in Figure 18 to demonstrate how BERT would tokenize a mention, tag, URL.

```

[21] 1 print(TOKENIZER.tokenize("#tothebone https://t.co/y7eizfybs8 @indiewire"))
      2
      [' ', 'to', '##the', '##bone', 'https', ':', '/', '/', 't', '.', 'co', '/', '/', 'y', '##7', '##ei', '##z', '##fy', '##bs', '##8', '@', 'indie', '##wire']
  
```

Figure 18. The tokenization of URLs, mentions, and tags by the BERT tokenizer

There are 2 famous techniques in NLP, namely, stemming and lemmatization. To decide which technique is more effective in mitigating the OOV issue, we had to test both experimentally. Stemming and lemmatization are common methods for getting the root or smaller version of a word. They differ in the application; lemmatization returns the roots of words, while stemming uses different rules to return different forms of the word. Stemming can be applied with different rules to reduce the suffix of words. A rule can be as simple as removing *s* from the end of the words; the word “kids” becomes “kid.” However, using these techniques with a pre-trained model like BERT did not improve the model's performance.

In the experiment, a setup was developed to aid in the development and testing of this method. To evaluate our methods, we carried out experiments using the mentioned libraries. Many techniques have been suggested to improve the representation of text. The technique presented is capable of producing a normalized dataset. Hence, the text will be represented in the most standardized way possible. Algorithm 2 shows the steps taken to extend contractions, correct spelling, and extend informal versions of gonna, wanna, etc.

To show an example from the dataset, we randomly selected 3 rows from the dataset, as shown in Figure 19. After adding the website, mention, and hashtag labels, the result is shown in Figure 19. The upper half of the figure shows the text before the addition of the labels, while the bottom half shows the resulting text after the addition of the label. The same process was followed for the other labels. BERT treats all 3 labels differently every time depending on the text accompanying the label. Because BERT uses the sub-words tokenization method, if the whole word is unknown, the texts will be divided to To show an example from the dataset, we randomly selected 3 rows from the dataset, as shown

in Figure 19. After adding the website, mention, and hashtag labels, the result is shown in Figure 19. The upper half of the figure shows the text before the addition of the labels, while the bottom half shows the resulting text after the addition of the label. The same

Algorithm 2. The steps of text normalization

Emojis dictionary:

Step 1: Load the dataset as a pandas dataframe

Step 2: Load cPickle library

Step 3: Import emojis dictionary

Step 4: Find all emojis with the dataframe and replace them with the corresponding key:value

Step 5: Check and count all emojis to test the effect

Contractions extension:

Step 1: Count the number of apostrophes

Step 2: Create a dictionary with common contractions

Step 3: Replace all matching contractions with their extended versions from the dictionary

Step 4: Recount the number of apostrophes for comparison

Spelling correction:

Step 1: Import the TextBlob library

Step 2: Apply TextBlob correct method on the dataframe

Step 3: Tokenize normalized version of the dataset with BERT tokenizer

Step 4: Move to phase 2 of normalization

URL custom label:

Step 1: Find URL within tweet text

Step 2: Add the "Website" label

Hashtag custom label:

Step 3: Find # within tweet text

Step 4: Add the "Hashtag" label

Mention custom label:

Step 5: Find @ within tweet text

Step 6: Add the "Mention" label

process was followed for the other labels. BERT treats all 3 labels differently every time depending on the text accompanying the label. Because BERT uses the sub-words tokenization method, if the whole word is unknown, the texts will be divided to make sub-

words tokens. Adding these labels will guarantee that BERT always treats the labels the same. As a result, this can be used as embedded features for BERT to consider. Previously, feature selection would be the option. However, with BERT, a text-embedded feature is the option. From our previous experimentations, we observed an improvement in BERT's performance when we had more tokens; no preprocessing or noise removal was applied. Hence, we are trying to enrich the text to give BERT more tokens to work with. However, the added tokens will hopefully be contextually more meaningful this time.

```
[ ] 1 print(df.loc[df.index[185732], 'tweet_text'])
    2 print(df.loc[df.index[263810], 'tweet_text'])
    3 print(df.loc[df.index[265304], 'tweet_text'])
```

How pissed are the gays gonna be when they find out that marriage sucks.

Made it halfway through this #MassachusettsSong nonsense: <http://www.bostonmagazine.com/news/blog/2013/10/22/ylvis-massachusetts-music-video/> ... They can't even say Havenhill right...

La Katana News is out! <http://paper.li/auroraferrer/1333415229> ... Stories via @fueradeseries @Cansaliebres_

```
[ ] 1 print(df1.loc[df1.index[185732], 'tweet_text'])
    2 print(df1.loc[df1.index[263810], 'tweet_text'])
    3 print(df1.loc[df1.index[265304], 'tweet_text'])
```

How pissed are the gays going to be when they find out that marriage sucks.

Made it halfway through this #MassachusettsSong nonsense: <http://www.bostonmagazine.com/news/blog/2013/10/22/ylvis-massachusetts-music-video/> ... They can't even say Havenhill right... Website Hashtag

La Katana News is out! <http://paper.li/auroraferrer/1333415229> ... Stories via @fueradeseries @Cansaliebres_ Website Mention

Figure 19. The effect of adding custom label

CHAPTER 4

RESULTS AND DISCUSSION

Fake News Spreaders Case Study (Esam & Leon, 2021)

Our best results for the Spanish and English datasets were 77% and 63%, respectively. Table 11 shows the results and the setup we used to achieve those results. We found that using BERT multilingual cased model with the determined hyperparameters in Table 11 for the Spanish language achieved the best results among all the three models, BERT, ROBERTA, and Longformer. The experiments on the English dataset showed that using the BERT base cased model with the determined hyperparameters in Table 11 achieved the best results among all different conducted experiments which differ in the type of models and the values of hyperparameters. In our experiments, we tried using all three models with different hyperparameters and different preprocessing techniques. Nevertheless, the best results were achieved when we did not apply any preprocessing technique with the determined hyperparameters. In both languages, the use of cased models always yielded better accuracies. ROBERTA and Longformer tokenizers can handle emojis that exist in most tweets nowadays. The problem we faced in using both models was when they converged and became stuck at the same starting loss and accuracy. Therefore, our choice fell on BERT because it started to converge in the early epochs. However, as mentioned earlier, the BERT tokenizer cannot handle emojis which creates a failure of using an important part of the data. The shortness of tweets is another factor that contributes to the low accuracy. In this task specifically, not all hate speech

spreaders' tweets are considered hateful. If a user is considered a hateful speech spreader, there might be only a few numbers of hateful tweets. As a result, this creates confusion for the model as the author is classified as a hate speech spreader, but not all his/her tweets are classified as hateful. We also found that some users repeat the same tweets more than on

Table 11. Models settings for the achieved results

Language	SPANISH	ENGLISH
SCORE	77%	63%
SCENARIO	1 st for training 3 rd for testing	1 st for training 3 rd for testing
PREPROCESSING	None	None
MODEL	bert-multilingual- cased	bert-base-cased
EPOCHS	50	50
BATCH SIZE	16	32
MAX LEN	60 tokens	50 tokens
LEARNING RATE	2 ⁻⁵	2 ⁻⁵

Text Pre-Processing Techniques Effect (Esam Alzahrani & Jololian, 2021)

In our study, we sought to examine the effect of the most commonly used preprocessing techniques on the gender profiling of authors when using a pretrained model: the BERT model (Esam Alzahrani & Jololian, 2021). To distinguish our five cases of preprocessing techniques from each other, we conducted an experiment for each case. Because transfer learning models were trained on a relatively large dataset, we thought that the pretrained models would perform better in downstream tasks when the downstream dataset was larger. After performing the five experiments, we found that the best case for BERT was not applying any preprocessing technique and that the worst was Case 4, when we applied five preprocessing techniques (i.e., mentions removal, retweet tags removal, hashtags removal, URLs removal, and punctuation removal). Removing stop words also negatively affected the use of the BERT model. The rest of the cases differed slightly in accuracy, as shown in Table 12. The difference between Case 4 and Case 5 contributed to a significant difference overall (i.e., approx. 8%). The fewer preprocessing techniques we applied, the higher accuracy we observed. However, the time needed to train the model for Case 5 was the longest. Table 13 shows that the same effect applies to the age profiling experiments. The case 5 where no pre-processing techniques were applied achieved the highest accuracy of 72%.

A possible explanation for those results is that pretrained models perform better on larger texts and need every token that they might learn from. Even though stop words might not be used as markers in machine learning methods (Reddy et al., 2017), the BERT model performed better when stop words were not removed. As mentioned, transfer learning techniques are features-independent, and their capability in contextually model language

makes them powerful enough to understand natural language. Our goal was to test aspects that can affect the performance of such pretrained models, namely the most commonly used preprocessing techniques for profiling the age and gender of authors. As a result, our study offers valuable findings that shed light on the best preprocessing techniques that can be applied when using a pretrained model to profile authors by gender.

Table 12. Results of gender profiling experiments. We consider using cross-validation to test the accuracy of the built models. We split the dataset into 90% for training and 10% for testing.

Case	Accuracy
Case 1	0.8229
Case 2	0.8074
Case 3	0.7946
Case 4	0.7886
Case 5	0.8667

Table 13. Results of age profiling experiments. We consider using cross-validation to test the accuracy of the built models. We split the dataset into 90% for training and 10% for testing.

Case	Accuracy
Case 1	0.68
Case 2	0.64
Case 3	0.40
Case 4	0.60
Case 5	0.72

Our tests revealed significant information about model selection and the manner in which certain hyperparameters are linked to a model's overall performance. Of several possible hyperparameters, we investigated learning rate and batch size, and we found that certain hyperparameter combinations had a significant effect on model accuracy and changing a hyperparameter value yielded drastically different model performances. It became clear that with some hyperparameters values the PTMs would not converge, which prevented model optimization in subsequent epochs. Finally, we observed that a model's learning rate is proportional to the extent to whether the model will converge or not attempting to adjust and optimize to achieve the optimal solution (E Alzahrani et al., 2022).

Table 14 presents the results of our experiments. We tested each model under the same conditions: A total of two classification tasks—age and gender prediction—were conducted for each model, and each model was tested with three different learning rates and three different batch sizes. Overall, the BERT model achieved the highest accuracy; moreover, BERT converged and optimized in each epoch to a greater extent than the other two models. When comparing RoBERTa and BERTweet, the latter achieved a higher level of accuracy and converged on more occasions than the former. Because the authors were divided into five age groups, the multi-class age-prediction task was more challenging than the binary-class gender-prediction task, and we observed that every model poorly predicted the authors' ages.

BERT achieved the highest accuracy rate of 87% for the gender-profiling task; the hyperparameter combination that enabled these results were a learning rate of 2^{-5} and a batch size of 32. As stated above, the age-profiling task was more challenging than the

gender-profiling task, and the models repeatedly as stated above, the age-profiling task was more challenging than the gender-profiling task, and the models repeatedly yielded poor performances; the highest accuracy rate of 76% was achieved by both BERT with a learning rate of 2^{-5} and a batch size of 16, and BERTweet with a learning rate of 3^{-5} and a batch size of 16. While we concluded that the effect of the batch size was less significant than that of the learning rate, batch size is not a negligible hyperparameter, because it still had a slight effect on the models' performances when the learning rate variable was controlled.

We noted that when the learning rate of each model was controlled, changing the batch size affected the model's performance. We ultimately concluded that BERT most accurately profiled the authors' ages with a learning rate of 2^{-5} and a batch size of 16, and

Table 14. Experiment results for different hyperparameters

		Age prediction accuracy			Gender prediction accuracy		
Model	Batch size	Learning rate			Learning rate		
		2^{-5}	3^{-5}	5^{-5}	2^{-5}	3^{-5}	5^{-5}
BERT	8	0.73	0.56	0.56	0.56	0.47	0.40
	16	0.76	0.56	0.56	0.86	0.5	0.53
	32	0.74	0.48	0.56	0.87	0.86	0.48
RoBERTa	8	0.51	0.56	0.56	0.56	0.35	0.35
	16	0.54	0.68	0.56	0.74	0.48	0.41
	32	0.55	0.56	0.56	0.79	0.5	0.35
BERTweet	8	0.35	0.56	0.56	0.56	0.48	0.40
	16	0.4	0.76	0.56	0.79	0.51	0.40
	32	0.59	0.56	0.56	0.79	0.49	0.40

most accurately profiled their genders with a learning rate of 2^{-5} and a batch size of 32; RoBERTa most accurately profiled ages with a learning rate of 3^{-5} and a batch size of 16, and genders with a learning rate of 2^{-5} and a batch size of 32; and BERTweet most accurately profiled ages with a learning rate of 3^{-5} learning rate and a batch size of 16, and genders with a learning rate of 2^{-5} and batch sizes of both 16 and 32. Finally, we observed that a 2^{-5} learning rate enabled the models to converge to a greater extent than the other learning-rate values. For some reason, the models could not converge with specific hyperparameters. The initial weights and values prevented the models from optimization. One explanation might be is that the model stuck in local minima. If we look at the problem as an optimization problem, that could be a reasonable explanation. In the effort to get a precise explanation, further investigation is needed.

The Analysis of Text Representation In Transfer Learning

The effect of our experimentation with different normalization methods is shown in Figures 20-25. Figure 20 shows the number of total tokens before we apply any normalization technique, while Figure 21 shows the total number of sub-word tokens (## tokens). Figures 22–25 show the effect on the tokens' numbers and lengths. The total number of tokens before the normalization was 10,351,5550 tokens. This number changed after translating the emojis in the dataset into texts, resulting in the total number of tokens increasing to 10,809,949. The BERT tokenizer can now benefit from 450k more tokens. To perform the emojis dictionary approach, we had to consider the PAN 2017 author profiling dataset. The obvious reason is that the dataset from PAN 2017 contains emojis and emoticons. Figures 22 and 23 show the statistics of tokens and sub-word tokens of the

dataset after the translation of emojis and emoticons.

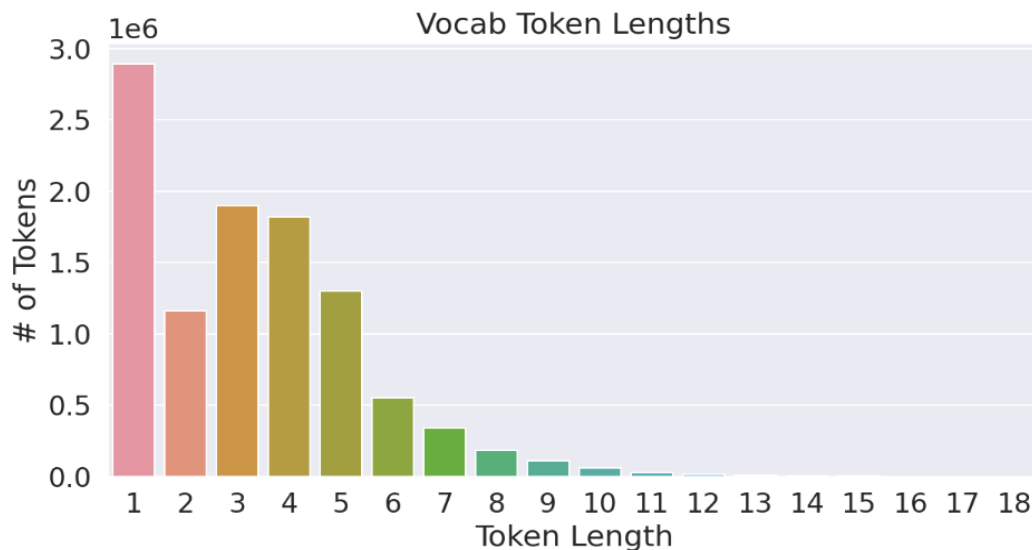


Figure 20. Vocabulary tokens length of PAN 17 dataset before emojis and emoticons translation

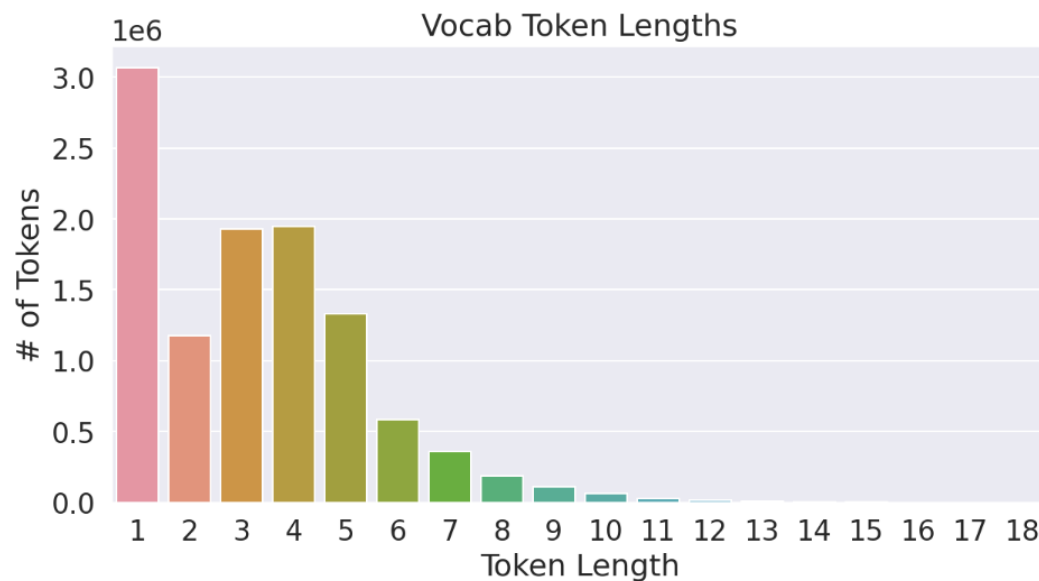


Figure 21. Vocabulary tokens length of PAN 17 dataset after emojis and emoticons translation

Figures 24 and 25 show the effect of contractions expansion after using our custom-built contraction expansion dictionary. As we can see from Figure 24, the number of single-character tokens decreased. This phenomenon resulted from the expansion of contractions

and informal words such as gonna and wanna. However, there is an increase in the sub-word tokens (##), as shown in Figure 25. The increase was on the non-single character sub-word tokens. This is still an acceptable addition, as non-single characters might be more beneficial to the BERT tokenizer and the text representation overall.

The last phase of our text normalization will be either lemmatizing or stemming the tweet texts. This will also result in reducing the sub-word tokens. We wanted to study that effect by running the same experiments when the text lemmatized or stemmed. The results show that stemming or lemmatizing the text was not a good idea with

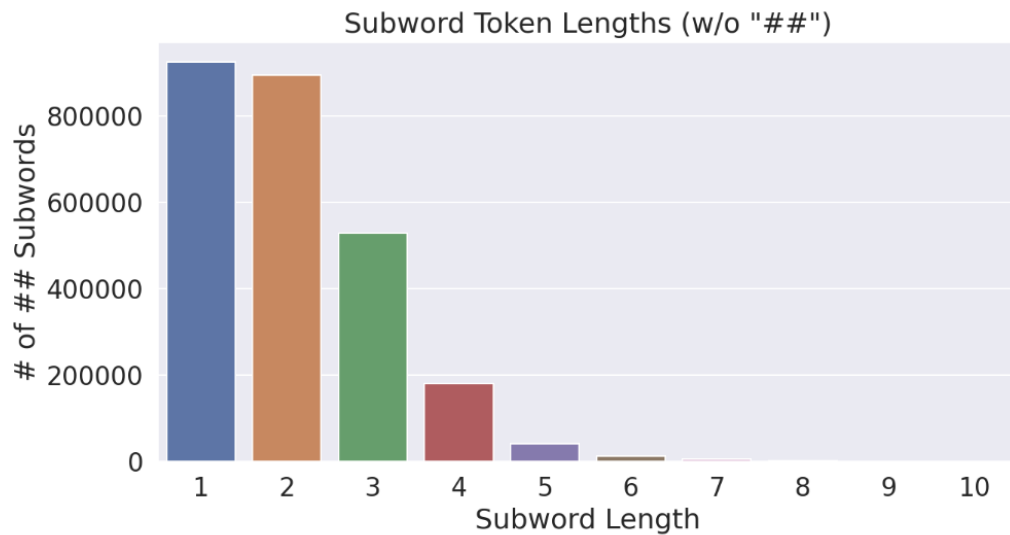


Figure 22. Sub-word tokens length of PAN 17 dataset before emojis and emoticons translation

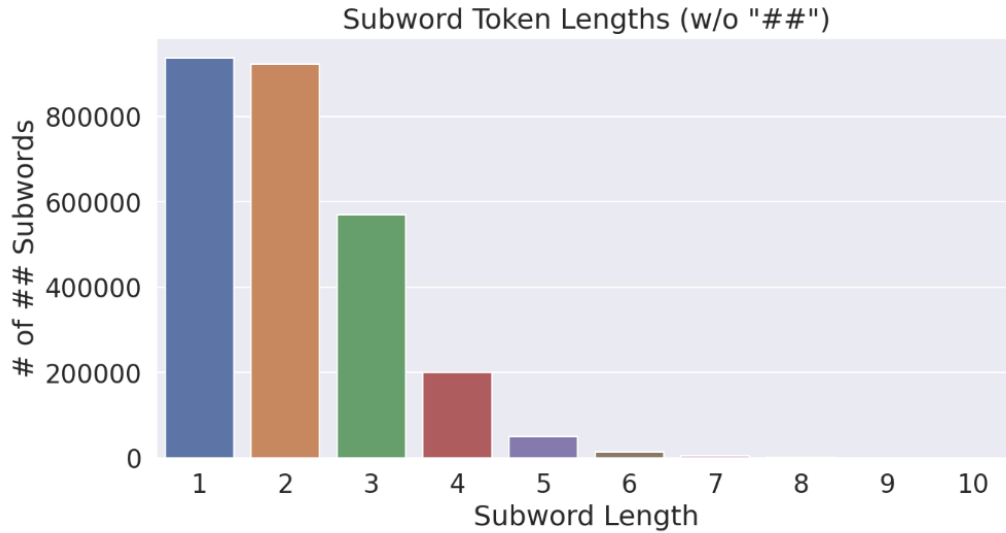


Figure 23. Sub-word tokens length of PAN 17 dataset after emojis and emoticons translation

BERT. The problem arises when the length of the whole dataset dramatically decreased. This will subsequently affect the number of generated tokens too. As mentioned, some solutions were effective with older approaches like machine and deep learning. However, they are not as effective when applied with BERT. This also proves the general behavior of BERT of performing well when more tokens are present, specifically when applied to profile the age and gender of authors.

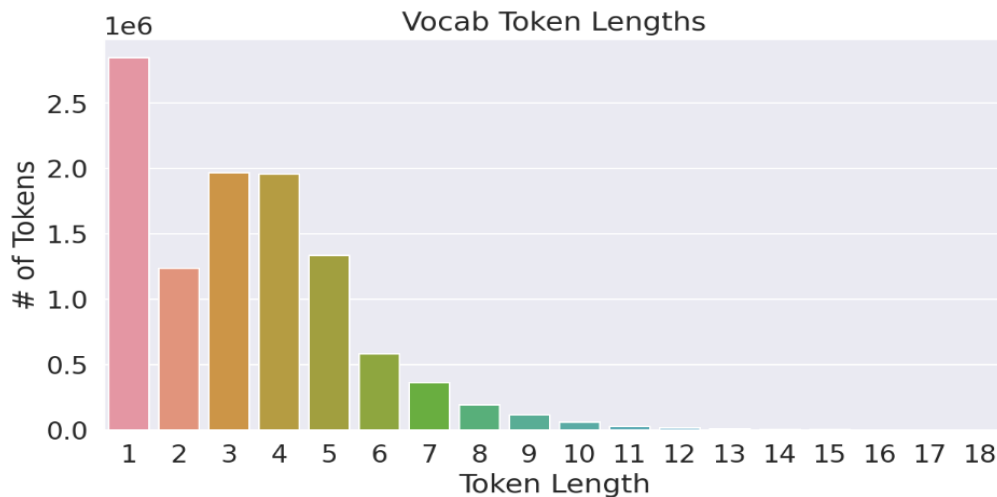


Figure 24. Vocabulary tokens length of PAN 17 dataset after emojis and emoticons translation and contractions expansion

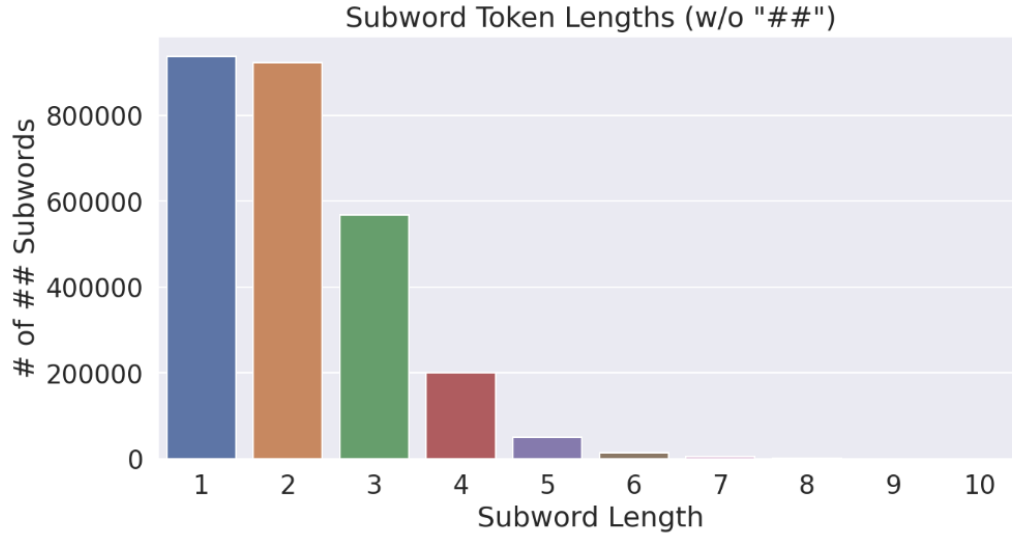


Figure 25. Sub-word tokens length of PAN 17 dataset after emojis and emoticons translation and contractions expansion

After performing all intended normalization processes, we again tested the performance of BERT in profiling the gender of authors using the dataset from PAN 2107. We first ran the model on the dataset without using any normalization techniques. Then, we ran BERT on the same dataset but after conducting 2 rounds of normalization. It is worth mentioning that we used the same hyperparameter values in all 3 different rounds. The first column in Table 15 shows the results before normalization. The middle column shows the result after applying the emojis dictionary. Finally, the far-right column shows the result after applying both the emojis dictionary and contractions expansion. It can be seen that the accuracy improved after the 2 normalization rounds, even though the improvement in the last round is less than the improvement achieved in the second round. However, there is always improvement, proving our initial hypothesis that manipulating and normalizing the dataset could result in better text representation and, eventually, better accuracy.

The outcome here is to prove that the concept of meaningful text enrichment based on the target task can lead to better results. Moreover, more techniques can be applied to further improve the accuracy. Researchers always look to optimize part of the architecture or change the whole pre-training process. However, we proved that simple changes to the text representation could benefit the powerful model greatly. This adds to the idea that understanding the domain to which one applies a model is essential to get the most out of that model.

It is known that BERT and its variants are powerful and perform well for NLP tasks. Nevertheless, some researchers argue about its ability for text classification tasks. We believe that paying *self-attention* to some details could change how we look at BERT for text classification. The answer is not always adding more tokens and messing up the generated weight from a very long pre-training process or re-pre-training BERT on a different dataset that is similar to or closer to one's target dataset. We will keep our effort to feed the BERT tokenizer the best possible representable dataset that can boost the performance of BERT or its variants.

Table 15. The results of gender profiling after text normalization

Epoch	Before	After emojis dictionary	After emojis dictionary and contraction expansion
1/4	Train loss 0.687 accuracy 0.539 Val loss 0.693 accuracy 0.507	Train loss 0.616 accuracy 0.666 Val loss 0.590 accuracy 0.700	Train loss 0.611 accuracy 0.672 Val loss 0.580 accuracy 0.712
2/4	Train loss 0.670 accuracy 0.598 Val loss 0.661 accuracy 0.622	Train loss 0.558 accuracy 0.741 Val loss 0.587 accuracy 0.711	Train loss 0.547 accuracy 0.752 Val loss 0.578 accuracy 0.723
3/4	Train loss 0.643 accuracy 0.640 Val loss 0.645 accuracy 0.643	Train loss 0.521 accuracy 0.785 Val loss 0.583 accuracy 0.722	Train loss 0.506 accuracy 0.800 Val loss 0.584 accuracy 0.723
4/4	Train loss 0.621 accuracy 0.662 Val loss 0.628 accuracy 0.655	Train loss 0.495 accuracy 0.814 Val loss 0.582 accuracy 0.726	Train loss 0.480 accuracy 0.8299 Val loss 0.580 accuracy 0.730

CHAPTER 5

CONCLUSION

As the use of transfer-learning techniques continues to evolve, PTMs such as BERT and its successors have been proven to perform most NLP tasks well, even though the task of text classification requires further attention and effort to achieve results that are comparable to other NLP domains. In our analysis, we explored the use of PTMs in the field of forensic authorship profiling, assessed different tokenization methods, and evaluated the effects of variables such as learning rate and batch size on model performance; our study lays the groundwork for the use of a transfer-learning technique to profile authored text, and our observations of the effects of the aforementioned variables will contribute to the extant literature and guide future research in this area.

Our experiments on how pre-processing techniques impact the gender profiling of authors when using a transfer learning model confirmed that the BERT performs best when no pre-processing techniques are applied. They also revealed that removing stop words lowers the accuracy by 1%. Those results indicate that pretrained models perform better when longer texts are present. Other common pre-processing techniques in the literature were included in the experiments and showed that they affect the pretrained model performance negatively. On top of that, our findings suggest that the use of pretrained models could be standardized, for it does not rely on many dependent parameters such as pre-processing and variable features.

The purpose of text normalization analysis is to demonstrate that meaningful text enrichment depending on the intended task can produce superior outcomes. In addition, further methods can be used to increase accuracy. Researchers are constantly trying to modify the pre-training procedure or maximize a certain aspect of the design. We did, however, show that the robust model may benefit significantly from making a few little adjustments to the text representation. This strengthens the notion that using a model effectively requires a grasp of the context to which it is applied.

In the course of our research, we identified some limitations. In addition to utilizing an unbalanced dataset in our experiments, we also only implemented three models for the sake of time and because of the limited scope of this study; including more models and processing the dataset would have yielded additional insights into this topic. In our future work, we will explore the possibility of employing more models to gain a deeper, more nuanced understanding of this subject, and we suspect that using different datasets could facilitate an acceptable generalization of the study findings.

We have always faced challenges dealing with the datasets such as the limited size, the repetitive tweets or words, and the amount of noise compared to the short tweets. In some cases, the markers are unclear as some words can have different meanings depending on the context or the user's intended meaning. Tokenizers play an important role in language analysis tasks.

For the future, the study has provided the groundwork for using transfer learning techniques to advance the field of author profiling, with findings that can serve as a starting point for using transfer learning in author profiling. Although the scope of the study was limited in terms of the pretrained model and the number of pre-processing techniques used,

future work could involve using more than one pretrained model in a bid to better generalize the findings. Researchers could also apply more pre-processing techniques to cover more pre-processing possibilities. Text representation or tokenization is directly linked to the performance of the model. Therefore, for future work, we would love to spend more time customizing the tokenizer we want to use to better represent the dataset. Moreover, we would consider spending more time exploring the tweets and eliminating any undesired noise. We would also consider using other pretrained models.

REFERENCES

- Adame-Arcia, Y., Castro-Castro, D., Bueno, R. O., & Muñoz, R. (2017). Author profiling, instance-based similarity classification: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings, 1866*.
- Aguilar, G., McCann, B., Niu, T., Rajani, N., Keskar, N., & Solorio, T. (2021). Char2Subword: Extending the Subword Embedding Space Using Robust Character Compositionality. *Findings of the Association for Computational Linguistics, Findings of ACL: EMNLP 2021*, 1640–1651. <https://doi.org/10.18653/v1/2021.findings-emnlp.141>
- Akhtyamova, L., Cardiff, J., & Ignatov, A. (2017). Twitter author profiling using word embeddings and logistic regression: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings, 1866*.
- Aleman, Y., Loya, N., Vilariño, D., & Pinto, D. (2013). Two methodologies applied to the author profiling task. *CEUR Workshop Proceedings, 1179*, 1–8.
- Altamimi, A., Alotaibi, S., & Alruban, A. (2019). *Surveying the Development of Authorship Identification of Text Messages Centre for Security , Communications and Network Research , Computer Sciences and Information Technology College , Majmaah University ,. 10(1)*, 953–966.
- Álvarez-Carmona, M. A., López-Monroy, A. P., Montes-Y-Gómez, M., Villaseñor-Pineda, L., & Escalante, H. J. (2015). INAOE’s participation at PAN’15: Author profiling task. *CEUR Workshop Proceedings, 1391*.
- Alzahrani, E, Qurashi, M. Al, & Jololian, L. (2022). Comparative Analysis of the Use of Pre-Trained Models to Profile Authors’ Ages and Genders. *2022 2nd International Conference on Computing and Machine Intelligence (ICMI)*, 1–7. <https://doi.org/10.1109/ICMI55296.2022.9873677>
- Alzahrani, Esam, & Jololian, L. (2021). How Different Text-preprocessing Techniques Using The {BERT} Model Affect The Gender Profiling of Authors. *CoRR, abs/2109.1*. <https://arxiv.org/abs/2109.13890>
- Argamon, S., Koppel, M., Fine, J., & Shimoni, A. R. (2003). Gender, genre, and writing style in formal written texts. *Text*, 23(3), 321–346. <https://doi.org/10.1515/text.2003.014>

- Argamon, S., Koppel, M., Pennebaker, J. W., & Schler, J. (2009). Automatically profiling the author of an anonymous text. *Communications of the ACM*, 52(2), 119–123. <https://doi.org/10.1145/1461928.1461959>
- Argamon, S., Whitelaw, C., Chase, P., Hota, S. R., Garg, N., & Levitan, S. (2007). Stylistic text classification using functional lexical features. *Journal of the American Society for Information Science and Technology*, 58(6), 802–822. <https://doi.org/10.1002/asi.20553>
- Arroju, M., Hassan, A., & Farnadi, G. (2015). Age, gender and personality recognition using tweets in a multilingual setting. *CEUR Workshop Proceedings*, 1391.
- Ashraf, S., Rizwan Iqbal, H., & Muhammad Adeel Nawab, R. (2016). Cross-Genre Author Profile Prediction Using Stylometry-Based Approach Notebook for PAN at CLEF 2016. *CEUR Workshop Proceedings*. <http://nlp.stanford.edu/software/tagger.shtml>
- Bagnall, D. (2015). Author identification using multi-headed recurrent neural networks. *CEUR Workshop Proceedings*, 1391.
- Bahdanau, D., Cho, K. H., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–15.
- Bakkar Deyab, R., Duarte, J., & Gonçalves, T. (2016). Author Profiling Using Support Vector Machines Notebook for PAN at CLEF 2016. *CEUR Workshop Proceedings*, 2–5.
- Bartoli, A., Dagri, A., Lorenzo, A. De, Medvet, E., & Tarlao, F. (2015). An Author Verification Approach Based on Differential Features Notebook for PAN at CLEF 2015. *Working Notes for CLEF 2015 Conference*, 1–7. <http://ceur-ws.org/Vol-1391/41-CR.pdf>
- Basile, A., Dwyer, G., Medvedeva, M., Rawee, J., Haagsma, H., & Nissim, M. (2017). N-GRAM: New groningen author-profiling model: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, 1866.
- Bayot, R., & Gonçalves, T. (2016). Author profiling using SVMs and Word embedding averages. *CEUR Workshop Proceedings*, 1609, 815–823.
- Bilan, I., & Zhekova, D. (2016). CAPS: A Cross-genre Author Profiling System Notebook for PAN at CLEF 2016. In *CEUR Workshop Proceedings*. <https://www.linkedin.com>
- Bouazizi, M., Beylerian, A., & Ohtsuki, T. (2017). *Submission to the 5th International Competition on Author Profiling*.
- Bougiatiotis, K., & Krithara, A. (2016). Author profiling using complementary second order attributes and stylometric features. *CEUR Workshop Proceedings*, 1609, 836–845.
- Burger, J. D., Henderson, J., Kim, G., & Zarrella, G. (2011). Discriminating gender on Twitter. *EMNLP 2011 - Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 1301–1309.

Busger, M., Carlotto, T., Kreutz, T., Medvedeva, M., Pool, C., Bjerva, J., Haagsma, H., & Nissim, M. (2016). GronUP: Groningen User Profiling Notebook for PAN. *CEUR Workshop Proceedings*.

Cheng, J., Dong, L., & Lapata, M. (2016). Long short-term memory-networks for machine reading. *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, 551–561. <https://doi.org/10.18653/v1/d16-1053>

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 1724–1734. <https://doi.org/10.3115/v1/d14-1179>

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. 1–9. <http://arxiv.org/abs/1412.3555>

Ciobanu, A. M., Zampieri, M., Malmasi, S., & Dinu, L. P. (2017). Including dialects and language varieties in author profiling: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, 1866.

Clément, L., Villemonte, É., & Clergerie, D. (2010). *MAF: a Morphosyntactic Annotation Framework*. June, 1–5. <ftp://ftp.inria.fr/INRIA/Projects/Atoll/Eric.Clergerie/LTC05.pdf>[http://file://localhost\(null\)%0Apapers3://publication/uuid/8E4AA805-3431-4066-BD29-6E62CF149FFD](http://file://localhost(null)%0Apapers3://publication/uuid/8E4AA805-3431-4066-BD29-6E62CF149FFD)

Clement, R., & Sharp, D. (2003). Ngram and Bayesian Classification of Documents for Topic and Authorship. *Literary and Linguistic Computing*, 18(4), 423–447. <https://doi.org/10.1093/lc/18.4.423>

Colombini, C., & Colella, A. (2011). Digital profiling: A computer forensics approach. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6908 LNCS, 330–343. https://doi.org/10.1007/978-3-642-23300-5_26

Cotterell, R., Mielke, S. J., Eisner, J., & Roark, B. (2018). Are all languages equally hard to language-model? *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2, 536–541. <https://doi.org/10.18653/v1/n18-2085>

Cruz, F. L., Haro, R. R., & Ortega, F. J. (2013). ITALICA at PAN 2013: An ensemble learning approach to author profiling: Notebook for PAN at CLEF 2013. *CEUR Workshop Proceedings*, 1179, 8–11.

D. Weren, E. R., Moreira, V. P., & de Oliveira, J. P. M. (2014). Exploring Information Retrieval features for Author Profiling---Notebook for PAN at CLEF 2014. In L. Cappellato, N. Ferro, M. Halvey, & W. Kraaij (Eds.), *Working Notes Papers of the CLEF 2014 Evaluation Labs*. CEUR-WS.org.

De-Arteaga, M., Jimenez, S., Dueñas, G., Mancera, S., & Baquero, J. (2013). Author profiling using corpus statistics, lexicons and stylistic features: Notebook for PAN at CLEF-2013. *CEUR Workshop Proceedings*, 1179.

de Valkeneer, H., & Mukhsinova, S. (2016). *Submission to the 4th International Competition on Author Profiling*.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference, 1*(Mlm), 4171–4186.

Díaz, A. A. C., & Hidalgo, J. M. G. (2013). Experiments with SMS translation and stochastic gradient descent in spanish text author profiling: Notebook for PAN at CLEF 2013. *CEUR Workshop Proceedings*, 1179.

Dichiu, D., & Rancea, I. (2016). Using Machine Learning Algorithms for Author Profiling In Social Media: Working Notes of CLEF 2016. *CEUR Workshop Proceedings*, 858–863. <http://ceur-ws.org/Vol-1609/16090858.pdf>

Don Kodiyan Florin Hardegger, S. N., & Cieliebak, M. (2017). Author Profiling with Bidirectional RNNs using Attention with GRUs. *CEUR Workshop Proceedings*.

Eder, M., Rybicki, J., & Kestemont, M. (2013). Stylometry with R: a suite of tools. *Digital Humanities 2013. Conference Abstracts*, 487–489. http://dh2013.unl.edu/abstracts/files/downloads/DH2013_conference_abstracts_print.pdf

El Boukkouri, H., Ferret, O., Lavergne, T., Noji, H., Zweigenbaum, P., & Tsujii, J. (2021). *CharacterBERT: Reconciling ELMo and BERT for Word-Level Open-Vocabulary Representations From Characters*. 6903–6915. <https://doi.org/10.18653/v1/2020.coling-main.609>

Esam, A., & Leon, J. (2021). Profiling Hate Speech Spreaders by classifying micro texts using BERT model: Notebook for PAN at CLEF 2021. *CEUR Workshop Proceedings*, 2936, 1796–1800.

F., B. (2002). “Delta”: A Measure of Stylistic Difference and a Guide to Likely Authorship. *Literary and Linguistic Computing*, 17, 267–287. <https://doi.org/10.1093/lc/17.3.267>

Flekova, L., & Gurevych, I. (2013). Can we hide in the web? Large scale simultaneous age and gender author profiling in social media: Notebook for PAN at CLEF 2013. *CEUR Workshop Proceedings*, 1179.

Frantzeskou, G., Stamatatos, E., Gritzalis, S., Chaski, C. E., & Howald, B. S. (2007). Identifying Authorship by Byte-Level N-Grams: The Source Code Author Profile (SCAP) Method. *International Journal of Digital Evidence*, 6(1), 1–18. <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Identifying+Authorshi>

p+by+Byte-Level+N-Grams:+The+Source+Code+Author+Profile+(SCAP)+Method#0

Frantzeskou, G., Stamatatos, E., Gritzalis, S., & Katsikas, S. (2006). Effective identification of source code authors using byte-level information. *Proceedings - International Conference on Software Engineering*, 2006, 893–896. <https://doi.org/10.1145/1134285.1134445>

Gamon, M. (2004). Linguistic correlates of style: authorship classification with deep linguistic analysis features. *{COLING} 2004: Proceedings of the 20th International Conference on Computational Linguistics*, 611–617. <https://www.aclweb.org/anthology/C04-1088>

Ganesh, B., Kumar M, A., & KP, S. (2017). *Submission to the 5th International Competition on Author Profiling*.

Gencheva, P., Boyanov, M., Deneva, E., Nakov, P., Kiprova, Y., Koychev, I., Georgiev, G., & Kliment, S. (2016). PANcakes Team : A Composite System of Genre-Agnostic Features For Author Profiling Notebook for PAN at CLEF 2016. *CEUR Workshop Proceedings*.

Giménez, M., Hernández, D. I., & Pla, F. (2015). Segmenting target audiences: Automatic author profiling using tweets. *CEUR Workshop Proceedings*, 1391.

Gómez-Adorno, H., Markov, I., Sidorov, G., Posadas-Durán, J. P., Sanchez-Perez, M. A., & Chanona-Hernandez, L. (2016). Improving Feature Representation Based on a Neural Network for Author Profiling in Social Media Texts. *Computational Intelligence and Neuroscience*, 2016. <https://doi.org/10.1155/2016/1638936>

González-gallardo, C. E., Montes, A., Sierra, G., Núñez-juárez, J. A., Salinas-lópez, A. J., & Ek, J. (2015). Tweets Classification Using Corpus Dependent Tags, Character and POS N-grams Notebook for PAN at CLEF 2015. *CLEF 2015 Labs and Workshops, Notebook Papers*.

Goswami, S., Sarkar, S., & Rustagi, M. (2009). Stylometric analysis of bloggers' age and gender. *Proceedings of the Third International ICWSM Conference, January 1999*, 214–217. <http://www.aaai.org/ocs/index.php/ICWSM/09/paper/viewPaper/208>

Grieve, J. (2007). Quantitative Authorship Attribution: An Evaluation of Techniques. *Literary and Linguistic Computing*, 22(3), 251–270. <https://doi.org/10.1093/lc/fqm020>

Grivas, A., Krithara, A., & Giannakopoulos, G. (2015). Author profiling using stylometric and structural feature groupings. *CEUR Workshop Proceedings*, 1391.

Halimi, A., & Ayday, E. (2017). *Profile Matching Across Unstructured Online Social Networks: Threats and Countermeasures*. <https://doi.org/10.5281/zenodo.1405662>

Harichandran, V. S., Breiting, F., Baggili, I., & Marrington, A. (2016). A cyber forensics needs analysis survey: Revisiting the domain's needs a decade later. *Computers and Security*, 57. <https://doi.org/10.1016/j.cose.2015.10.007>

- Hedegaard, S., & Simonsen, J. G. (2011). Lost in Translation: Authorship Attribution using Frame Semantics. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 65–70. <https://www.aclweb.org/anthology/P11-2012>
- Hirst, G. (2007). Bigrams of syntactic labels for authorship discrimination of short texts. *Literary and Linguistic Computing*.
- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Hoorn, J. F., Frank, S. L., Kowalczyk, W., & van der Ham, F. (1999). Neural network identification of poets using letter sequences. *Literary and Linguistic Computing*, 14(3), 311–338. <https://doi.org/10.1093/lc/14.3.311>
- Huggingface. (n.d.). *Summary of the tokenizers*. Huggingface.
- Iqbal, H. R., Ashraf, M. A., Muhammad, R., & Nawab, A. (2015). Predicting an author's demographics from text using Topic Modeling approach Notebook for PAN at CLEF 2015. *CLEF 2015 Labs and Workshops, Notebook Papers*.
- Jankowska, M., Kešelj, V., & Milios, E. (n.d.). *CNG text classification for authorship profiling task Notebook for PAN at CLEF 2013*. <http://www.cs.dal.ca/>
- Joachims, T. (2002). Learning To Classify Text Using Support Vector Machines. In *Fraunhofer AIS* (Vol. 668). <https://doi.org/10.1007/978-1-4615-0907-3>
- Jurafsky, D., & Martin, J. H. (2009). *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc.
- Keretna, S., Hossny, A., & Creighton, D. (2013). Recognising user identity in twitter social networks via text mining. *Proceedings - 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013*, 3079–3082. <https://doi.org/10.1109/SMC.2013.525>
- Khan, J. A. (2017). Author profile prediction using trend and word frequency based analysis in text: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, 1866.
- Kheng, G., Laporte, L., & Granitzer, M. (2017). INSA Lyon and UNI passau's participation at PAN@CLEF'17: Author Profiling task: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, 1866.
- Kim, Y., Denton, C., Hoang, L., & Rush, A. M. (2017). Structured attention networks. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 1–21.
- Kim, Y., Jernite, Y., Sontag, D., & Rush, A. M. (2016). Character-Aware neural language models. *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, 2741–2749. <https://doi.org/10.1609/aaai.v30i1.10362>

- Kiprov, Y., Hardalov, M., Nakov, P., & Koychev, I. (2015). SU@PAN'2015: Experiments in author profiling. *CEUR Workshop Proceedings, 1391*. <http://www.wdyl.com/>
- Kocher, M. (2015). UniNE at CLEF 2015: Author Profiling. *CEUR Workshop Proceedings, 1391*. <http://jetscram.com/blog/industry-news/social-media-user-statistics-and-age-demographics->
- Kocher, M., & Savoy, J. (2016). UniNE at CLEF 2016: Author profiling. *CEUR Workshop Proceedings, 1609*, 903–911.
- Kocher, M., & Savoy, J. (2017). Unine at CLEF 2017: Author profiling reasoning: Notebook for pan at CLEF 2017. *CEUR Workshop Proceedings, 1866*.
- Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., & Herbst, E. (2007). {M}oses: Open Source Toolkit for Statistical Machine Translation. *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, 177–180. <https://aclanthology.org/P07-2045>
- Koppel, M., Argamon, S., & Shimon, A. R. (2002). *categorizingTextByGender*. 17(4), 401–412.
- Koppel, M., & Winter, Y. (2013). Determining If Two Documents Are Written by the Same Author. *Journal of the American Society for Information Science and Technology*, 64(July), 1852–1863. <https://doi.org/10.1002/asi>
- Kuchaiev, O., & Ginsburg, B. (2019). Factorization tricks for LSTM networks. *5th International Conference on Learning Representations, ICLR 2017 - Workshop Track Proceedings*, 1–6.
- Layton, R., McCombie, S., & Watters, P. (2012). Authorship Attribution of IRC Messages Using Inverse Author Frequency. *2012 Third Cybercrime and Trustworthy Computing Workshop*, 7–13. <https://doi.org/10.1109/CTC.2012.11>
- Layton, Robert, Watters, P., & Dazeley, R. (2010). Authorship attribution for Twitter in 140 characters or less. *Proceedings - 2nd Cybercrime and Trustworthy Computing Workshop, CTC 2010*, 1–8. <https://doi.org/10.1109/CTC.2010.17>
- Lim, W.-Y., Goh, J., & Thing, V. L. L. (n.d.). *Content-centric age and gender profiling Notebook for PAN at CLEF 2013*.
- Lin, Z., Feng, M., Dos Santos, C. N., Yu, M., Xiang, B., Zhou, B., & Bengio, Y. (2017). A structured self-attentive sentence embedding. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 1–15.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*.

1. <http://arxiv.org/abs/1907.11692>

López-Monroy, A. P., Montes-Y-gómez, M., Escalante, H. J., & Villaseñor-Pineda, L. (2014). Using intra-profile information for author profiling: Notebook for PAN at CLEF 2014. *CEUR Workshop Proceedings, 1180*, 1116–1120.

López-monroy, A. P., Montes-y-gómez, M., Escalante, H. J., Villaseñor-pineda, L., & Villatoro-tello, E. (2013). *INAOE ' s participation at PAN ' 13 : Author Profiling task Notebook for PAN at CLEF 2013*.

Lundeqvist, E., & Svensson, M. (2017). *Author profiling: A machine learning approach towards detecting gender, age, and native language of users in social media*. 17013, 81.

Ma, W., Cui, Y., Si, C., Liu, T., Wang, S., & Hu, G. (2020). *CharBERT : Character-aware Pre-trained Language Model*. 39–50.

Maharjan, S., Shrestha, P., & Solorio, T. (2014). *A Simple Approach to Author Profiling in MapReduce*. 1121–1128.

Maharjan, S., & Solorio, T. (2015). Using wide range of features for author profiling. *CEUR Workshop Proceedings, 1391*.

Maitra, P., Ghosh, S., & Das, D. (2015). Authorship Verification – An Approach based on Random Forest Notebook for PAN at CLEF 2015. *Working Notes for CLEF 2015 Conference*, 1–9. <http://ceur-ws.org/Vol-1391/134-CR.pdf>

Malte, A., & Ratadiya, P. (2019). *Evolution of transfer learning in natural language processing*. <http://arxiv.org/abs/1910.07370>

Mamgain, S., Balabantaray, R. C., & Das, A. K. (2019). Author profiling: Prediction of gender and language variety from document. *Proceedings - 2019 International Conference on Information Technology, ICIT 2019*, 473–477. <https://doi.org/10.1109/ICIT48102.2019.00089>

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Markov, I., Gómez-Adorno, H., & Sidorov, G. (2017). Language- and subtask-dependent feature selection and classifier parameter tuning for author Profiling: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings, 1866*.

Markov, I., Gómez-Adorno, H., Sidorov, G., & Gelbukh, A. F. (2016). Adapting Cross-Genre Author Profiling to Language and Corpus: Working Notes of CLEF 2016. *CEUR Workshop Proceedings, 947–955*. <http://ceur-ws.org/Vol-1609/16090947.pdf>

Marquardt, J., Farnadi, G., Vasudevan, G., Davalos, S., Teredesai, A., & Cock, M. De. (n.d.). *Age and Gender Identification in Social Media*.

- Martinc, M., Skrjanec, I., Zupan, K., & Pollak, S. (2017). PAN 2017: Author Profiling-Gender and Language Variety Prediction. *Working Notes of CLEF 2017-Conference and Labs of the Evaluation Forum, Ireland, 11-14 September*.
- McCollister, C., Huang, S., & Luo, B. (2015). Building topic models to predict author attributes from Twitter messages. *CEUR Workshop Proceedings*, 1391.
- Mechti, S., Jaoua, M., & Belguith, L. H. (2014). Machine learning for classifying authors of anonymous tweets, blogs, reviews and social media: Notebook for PAN at CLEF 2014. *CEUR Workshop Proceedings*, 1180, 1137–1142.
- Meina, M., Brodzińska, K., Celmer, B., Czoków, M., Patera, M., Pazecki, J., & Wilk, M. (2013). Ensemble-based classification for author profiling using various features. *PAN - Uncovering Plagiarism, Authorship, and Social Software Misuse a Benchmarking Activity on Uncovering Plagiarism, Authorship and Social Software Misuse*. <http://www.uni-weimar.de/medien/webis/research/events/pan-13/pan13-papers-final/pan13-author-profiling/meina13-notebook.pdf>
- Miculicich Werlen, L. (2015). Statistical Learning Methods for Profiling Analysis Notebook for PAN at CLEF 2015. *CLEF 2015 Labs and Workshops, Notebook Papers*.
- Mielke, Sabrina J., Alyafeai, Z., Salesky, E., Raffel, C., Dey, M., Gallé, M., Raja, A., Si, C., Lee, W. Y., Sagot, B., & Tan, S. (2021). *Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP*. <http://arxiv.org/abs/2112.10508>
- Mielke, Sebastian J., Cotterell, R., Gorman, K., Roark, B., & Eisner, J. (2020). What kind of language is hard to language-model? *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 4975–4989. <https://doi.org/10.18653/v1/p19-1491>
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 1–12.
- Miura, Y., Taniguchi, T., Taniguchi, M., & Ohkuma, T. (2017). Author profiling with word+character neural attention network: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, 1866.
- Modaresi, P., Liebeck, M., & Conrad, S. (2016). Exploring the effects of cross-genre machine learning for author profiling in PAN 2016. *CEUR Workshop Proceedings*, 1609, 970–977.
- Moreau, E., & Vogel, C. (2013). *Style-based distance features for author profiling*.
- Najib, F., Cheema, W. A., & Nawab, R. M. A. (2015). Author's traits prediction on Twitter data using content based approach. *CEUR Workshop Proceedings*, 1391. <http://pan.webis.de/>

- Nguyen, D., Gravel, R., Trieschnigg, D., & Meder, T. (2013). *Edinburgh Research Explorer “ How Old Do You Think I Am ? ” A Study of Language and Age in Twitter “ How Old Do You Think I Am ? ”: A Study of Language and Age in Twitter*. 439–448.
- Nguyen, D. Q., Vu, T., & Tuan Nguyen, A. (2020). *BERTweet: A pre-trained language model for English Tweets*. 9–14. <https://doi.org/10.18653/v1/2020.emnlp-demos.2>
- Nowson, S., Perez, J., Brun, C., Mirkin, S., & Roux, C. (2015). XRCE personal language analytics engine for multilingual author profiling. *CEUR Workshop Proceedings*, 1391.
- Ogaltsov, A., & Romanov, A. (2017). Language variety and gender classification for Author Profiling in PAN 2017: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, 1866.
- Oliveira, W., Justino, E., & de Oliveira, L. (2013). Comparing compression models for authorship attribution. *Forensic Science International*, 228, 100–104. <https://doi.org/10.1016/j.forsciint.2013.02.025>
- Pacheco, M. L., Fernandes, K., & Porco, A. (2015). Random forest with increased generalization: A universal background approach for authorship verification. *CEUR Workshop Proceedings*, 1391.
- Panigrahi, S., Nanda, A., & Swarnkar, T. (2021). A Survey on Transfer Learning. *Smart Innovation, Systems and Technologies*, 194, 781–789. https://doi.org/10.1007/978-981-15-5971-6_83
- Parikh, A., Täckström, O., Das, D., & Uszkoreit, J. (2016). A Decomposable Attention Model for Natural Language Inference. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2249–2255. <https://doi.org/10.18653/v1/D16-1244>
- Pastor López-Monroy, A., Montes-Y-Gómez, M., Escalante, H. J., Villaseñor-Pineda, L., & Solorio, T. (2017). Social-media users can be profiled by their similarity with other users: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, 1866.
- Patra, B. G., Banerjee, S., Das, D., & Saikh, T. (2013). *Automatic Author Profiling Based on Linguistic and Stylistic Features*. 1–8.
- Paulus, R., Xiong, C., & Socher, R. (2018). A deep reinforced model for abstractive summarization. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, i, 1–12.
- Pavan, A., Mogadala, A., & Varma, V. (2013). *Author profiling using LDA and Maximum Entropy*.
- Peng, F., Schuurmans, D., & Wang, S. (2004). Augmenting Naive Bayes Classifiers with Statistical Language Models. *Information Retrieval*, 7(3), 317–345. <https://doi.org/10.1023/B:INRT.0000011209.19643.e2>

- Pervaz, I., Ameer, I., Sittar, A., & Nawab, R. M. A. (2015). Identification of author personality traits using stylistic features. *CEUR Workshop Proceedings*, 1391.
- Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1, 2227–2237. <https://doi.org/10.18653/v1/n18-1202>
- Pimas, O., Rexha, A., Kröll, M., & Kern, R. (2016). Profiling microblog authors using concreteness and sentiment. *CEUR Workshop Proceedings*, 1609, 978–983.
- Pinter, Y. (2021). *Integrating Approaches to Word Representation*. September, 1–22. <http://arxiv.org/abs/2109.04876>
- Posadas-Durán, J. P., Markov, I., Gómez-Adorno, H., Sidorov, G., Batyrshin, I., Gelbukh, A., & Pichardo-Lagunas, O. (2015). Syntactic N-grams as features for the author profiling task. *CEUR Workshop Proceedings*, 1391. <http://www.cic.ipn.mx/~sidorov>
- Poulston, A., Stevenson, M., & Bontcheva, K. (2015). Topic Models and n-gram Language Models for Author Profiling Notebook for PAN at CLEF 2015. *CLEF 2015 Labs and Workshops, Notebook Papers*.
- Przybyła, P., & Teisseyre, P. (2015). What do your look-alikes say about you? Exploiting strong and weak similarities for author profiling. *CEUR Workshop Proceedings*, 1391.
- Qian, T., Liu, B., Chen, L., Peng, Z., Zhong, M., He, G., Li, X., & Xu, G. (2016). Tri-Training for authorship attribution with limited training data: A comprehensive study. *Neurocomputing*, 171, 798–806. <https://doi.org/10.1016/j.neucom.2015.07.064>
- Qiu, X. P., Sun, T. X., Xu, Y. G., Shao, Y. F., Dai, N., & Huang, X. J. (2020). Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10), 1872–1897. <https://doi.org/10.1007/s11431-020-1647-3>
- Quick, D., Martini, B., & Choo, K.-K. R. (2014). Cloud Storage Forensics. *Cloud Storage Forensics*, October. <https://doi.org/10.1016/B978-0-12-419970-5.00002-8>
- Rammstedt, B., & John, O. P. (2007). Measuring personality in one minute or less: A 10-item short version of the Big Five Inventory in English and German. *Journal of Research in Personality*, 41(1), 203–212. <https://doi.org/https://doi.org/10.1016/j.jrp.2006.02.001>
- Rangel, F., Celli, F., Rosso, P., Potthast, M., Stein, B., & Daelemans, W. (2015). Overview of the 3rd Author Profiling Task at PAN 2015. *CEUR Workshop Proceedings*, 1179. <http://pan.webis.de>
- Rangel, F., Celli, F., Rosso, P., Potthast, M., Stein, B., Daelemans, W., Cappellato, L., Ferro, N., Jones, G., & San Juan, E. (n.d.). *Overview of the 3rd Author Profiling Task at PAN 2015* (Vol. 1391). <http://pan.webis.de>

- Rangel, F., & Chugur, I. (2014). *Overview of the 2nd Author Profiling Task at PAN 2014*.
- Rangel, F., Rosso, P., Koppel, M., Stamatatos, E., & Inches, G. (2013a). Overview of the author profiling task at PAN 2013. *CEUR Workshop Proceedings*, 1179.
- Rangel, F., Rosso, P., Koppel, M., Stamatatos, E., & Inches, G. (2013b). Overview of the author profiling task at PAN 2013. *CEUR Workshop Proceedings*, 1179, 8–11.
- Rangel, F., Rosso, P., Potthast, M., & Stein, B. (2017). Overview of the 5th Author Profiling Task at PAN 2017: Gender and Language Variety Identification in Twitter. *CEUR Workshop Proceedings*, 2380.
- Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., & Stein, B. (2016). Overview of the 4th author profiling task at PAN 2016: Cross-genre evaluations. *CEUR Workshop Proceedings*, 1609, 750–784.
- Reddy, T. R., Vardhan, B. V., & Reddy, P. V. (2017). N-gram approach for gender prediction. *Proceedings - 7th IEEE International Advanced Computing Conference, IACC 2017*, 860–865. <https://doi.org/10.1109/IACC.2017.0176>
- Rocha, A., Scheirer, W. J., Forstall, C. W., Cavalcante, T., Theophilo, A., Shen, B., Carvalho, A. R. B., & Stamatatos, E. (2017). Authorship Attribution for Social Media Forensics. *IEEE Transactions on Information Forensics and Security*, 12(1), 5–33. <https://doi.org/10.1109/TIFS.2016.2603960>
- Sagot, B., & Boullier, P. (2008). SxPipe 2 : architecture pour le traitement présyntaxique de corpus bruts. *Traitement Automatique Des Langues*, 49(2), 155–188.
- Santosh, K., Bansal, R., Shekhar, M., & Varma, V. (2013). Author profiling: Predicting age and gender from blogs: Notebook for PAN at CLEF 2013. *CEUR Workshop Proceedings*, 1179, 23–27.
- Sapkota, U., & Solorio, T. (2013). *Author Profiling for English and Spanish Text Notebook for PAN at CLEF 2013*. 2011, 2011–2014.
- Savoy, J. (2012). Authorship Attribution Based on Specific Vocabulary. *ACM Trans. Inf. Syst.*, 30(2). <https://doi.org/10.1145/2180868.2180874>
- Savoy, J. (2013). The federalist papers revisited: A collaborative attribution scheme. *Proceedings of the ASIST Annual Meeting*, 50(1). <https://doi.org/10.1002/meet.14505001036>
- Schaetti, N. (2017). UniNE at CLEF 2017: TF-IDF and Deep-Learning for author profiling: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, 1866.
- Schaetti, N., & Savoy, J. (2018). *Comparison of Neural Models for Gender Profiling*. June.
- Schler, J., Koppel, M., Argamon, S., & Pennebaker, J. (2006). *Effects of Age and Gender*

on Blogging.

Seelam, S. S. R., Kumar, S., Gopi, C. M., & Raghunadha, R. T. (2018). A New Term Weight Measure for Gender and Age Prediction of the Authors by analyzing their Written Texts. *Proceedings of the 8th International Advance Computing Conference, IACC 2018*, 150–156. <https://doi.org/10.1109/IADCC.2018.8692092>

Sennrich, R., Haddow, B., & Birch, A. (2016). Neural machine translation of rare words with subword units. *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers, 3*, 1715–1725. <https://doi.org/10.18653/v1/p16-1162>

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., & Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 1–19.

Siang, L. Y., & Thing, V. L. L. (2014). *Submission to the Author Profiling Task from the Institute for Infocomm Research, Singapore*.

Sierra, S., Montes-Y-gómez, M., Solorio, T., & González, F. A. (2017). Convolutional neural networks for author profiling: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings, 1866*.

Stamatatos, E., Fakotakis, N., & Kokkinakis, G. (2000). Automatic Text Categorization in Terms of Genre and Author. *Computational Linguistics*, 26(4), 471–495. <https://doi.org/10.1162/089120100750105920>

Stolerman, A., Overdorf, R., Afroz, S., & Greenstadt, R. (2014). Breaking the closed-world assumption in stylometric authorship attribution. *IFIP Advances in Information and Communication Technology*, 433, 185–205. https://doi.org/10.1007/978-3-662-44952-3_13

Şulea, O. M., & Dichiu, D. (2015). Automatic profiling of Twitter users based on their tweets. *CEUR Workshop Proceedings, 1391*.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 4(January), 3104–3112.

Tellez, E. S., Miranda-Jiménez, S., Graff, M., & Moctezuma, D. (2017). Gender and language-variety identification with MicroTC: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings, 1866*.

Ucelay, M. J. G., Villegas, M. P., Funez, D. G., Cagnina, L. C., Errecalde, M. L., Ramirez-De-La-Rosa, G., & Villatoro-Tello, E. (2016). Profile-based approach for age and gender identification. *CEUR Workshop Proceedings, 1609*, 864–873.

van Halteren, H. (2004). *Linguistic profiling for author recognition and verification*. 199-es. <https://doi.org/10.3115/1218955.1218981>

- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 2017-Decem(Nips)*, 5999–6009.
- Villena-román, J., & González-cristóbal, J. C. (2014). *DAEDALUS at PAN 2014 : Guessing Tweet Author ' s Gender and Age*.
- Warikoo, A. (2014). Proposed Methodology for Cyber Criminal Profiling. *Information Security Journal*, 23, 172–178. <https://doi.org/10.1080/19393555.2014.931491>
- Yang, P., & Chen, Y. (2018). A survey on sentiment analysis by using machine learning methods. *Proceedings of the 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2017, 2018-Janua*, 117–121. <https://doi.org/10.1109/ITNEC.2017.8284920>
- Yang, Q., Zhang, Y., Dai, W., & Pan, S. J. (2020). Transfer Learning in Natural Language Processing. *Transfer Learning*, 234–256. <https://doi.org/10.1017/9781139061773.020>
- Yepes, Ray (ATX Forensics LLC, Austin, T. (2016). *The Art of Profiling in a Digital World.pdf* (p. 8). International Association of Chiefs of Police.
- Zhao, Y., & Zobel, J. (2005). Effective and scalable authorship attribution using function words. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3689 LNCS, 174–189. https://doi.org/10.1007/11562382_14
- Zhao, Y., & Zobel, J. (2007). Entropy-Based Authorship Search in Large Document Collections. *Proceedings of the 29th European Conference on IR Research*, 381–392

APPENDIX A

Literature Review Features And Different Studies Comparison

Table 16 The variations of features

Feature
1. Number of exact words matches
2. Number of unique forms of matching stems
3. Number of unique forms of matching words
4. Mean frequency of word matches
5. Minimum frequency of word matches
6. Sum of inverse frequencies of word matches
7. Mean frequency of all words
8. Minimum frequency of all words
9. Sum of inverse frequencies of all words
10. Mean tf-idf scores of matching words
11. Sum of tf-idf scores of matching words
12. Max of tf-idf scores of matching words
13. Sum of tf-idf score of all words
14. Max of tf-idf score of all words
15. Distance between two furthest matching words
16. Combined distance between two furthest matching words
17. Distance between the lowest frequency words
18. Distance between the two highest tf-idf frequency words
19. Fraction of matching character-level unigram out of total
20. Fraction of matching character-level bigram out of total
21. Similarity of bigram frequencies between two strings

APPENDIX B

The Extended Version of Literature Review Studies Comparison

Table 17: The extended version of Literature review studies comparison

Paper	Accuracy	Preprocessing	feature	classifier
(Aleman et al., 2013)	0.3292	N/A	<ul style="list-style-type: none"> • Stylistic features (punctuation marks, capital letters, quotations, etc.) • POS tags • Readability • 	<ul style="list-style-type: none"> • Random Forest
(Iqbal et al., 2015)	0.5854	<ul style="list-style-type: none"> • HTML code removal 	<ul style="list-style-type: none"> • Topic modeling with LDA • 	<ul style="list-style-type: none"> • Random Forest (classification), J48 (regression)
(Ashraf et al., 2016)	0.2564	<ul style="list-style-type: none"> • HTML and XML codes removal • Feature selections 	<ul style="list-style-type: none"> • Stylistic features (unique words, sentiment words, etc.) • Stylistic features with n-grams • Stylistic with POS • Vocabulary richness • 	<ul style="list-style-type: none"> • Different tree-based algorithm (e.g. Random Forest, J48, LADTree)
(Pimas et al., 2016)	0.1410	N/A	<ul style="list-style-type: none"> • Stylistic features (unique words, sentiment words, etc.) • correctness 	<ul style="list-style-type: none"> • Different tree-based algorithm (e.g. Random Forest, J48, LADTree)
(Bouazizi et al., 2017)	0.2479	N/A	<ul style="list-style-type: none"> • Discriminative words (e.g., slang, locations, brands, stylistic patterns) 	<ul style="list-style-type: none"> • Random Forest

(Miculicich Werlen, 2015)	0.7115	N/A	<ul style="list-style-type: none"> • Punctuation marks • Topic modeling with LDA • LIWC 	<ul style="list-style-type: none"> • Linear Discriminant Analysis (regression)
(McCollister et al., 2015)	0.6746	N/A	<ul style="list-style-type: none"> • Topic modeling with LDA 	<ul style="list-style-type: none"> • Rotation Forest • Bagging (regression)
(Giménez et al., 2015)	0.5917	N/A	<ul style="list-style-type: none"> • Character flooding • Letter case • Twitter-specific features (links, hashtags, mentions) • Word n-grams • TF-IDF n-grams • Polarity words, emotions • NRC 	<ul style="list-style-type: none"> • Linear regression
(Flekova & Gurevych, 2013)	0.2785	<ul style="list-style-type: none"> • Chunk of training data for dimensionality reduction 	<ul style="list-style-type: none"> • Stylistic features (punctuation marks, capital letters, quotations, etc.) • Readability • Content features (Latent Semantic Analysis, bag of words, TF-IDF, dictionary-based words, topic-based words, entropy-based words, etc.) • Named Entities • Emotion words • 	<ul style="list-style-type: none"> • Logistic regression
(De-Arteaga et al., 2013)	0.2450	N/A	<ul style="list-style-type: none"> • Stylistic features (punctuation marks, capital letters, quotations, etc.) • 	<ul style="list-style-type: none"> • Logistic regression

liau14 (Siang & Thing, 2014)	0.26785	N/A	<ul style="list-style-type: none"> • The occurrence of emotions • n-grams • bag of words • Specific gender phrases 	<ul style="list-style-type: none"> • Logistic regression
(Maharjan et al., 2014)	0.241125		<ul style="list-style-type: none"> • Punctuation marks • The occurrence of emotions • n-grams, bag of words • Second-order representation 	<ul style="list-style-type: none"> • Logistic regression
(D. Weren et al., 2014)	0.227175	<ul style="list-style-type: none"> • HTML and XML cleaning • Escaped invalid characters 	<ul style="list-style-type: none"> • HTML tags (img,href,br) • Flesch-Kinkaid • Information retrieval (cosine similarity, Okapi BM25) 	<ul style="list-style-type: none"> • Logistic regression
(Maharjan & Solorio, 2015)	0.6623	<ul style="list-style-type: none"> • Hashtags, mentions 	<ul style="list-style-type: none"> • Question marks • Character n-grams • Topic modeling with LDA • Gender-specific phrases (my wife, my husband) 	<ul style="list-style-type: none"> • Logistic Regression
(Modaresi et al., 2016)	0.3846	N/A	<ul style="list-style-type: none"> • Stylistic features (unique words, sentiment words, etc.) • Stylistic features with n-grams 	<ul style="list-style-type: none"> • Logistic regression

				•	
(Bilan & Zhekova, 2016)	0.3333	<ul style="list-style-type: none"> • HTML and XML codes removal • Hashtags, RTs, URLs 	<ul style="list-style-type: none"> • Stylistic features (unique words, sentiment words, etc.) • Stylistic features with n-grams • Stylistic with POS • Collocations and LDA 	• Logistic regression	
(Martinc et al., 2017)	0.7042	<ul style="list-style-type: none"> • HTML and XML codes removal • Stop words removal • Punctuation signs removal 	<ul style="list-style-type: none"> • Character and word n-gram • Specific list of words for language variety • Characters flooding • Emotion or/and language expression • Emojis and sentiment words 	• Logistic regression	
(Ogaltsov & Romanov, 2017)	0.645	N/A	<ul style="list-style-type: none"> • Character and word n-gram • High order character n-grams 	• Logistic regression	
(Akhtyamova et al., 2017)	0.4333	N/A	<ul style="list-style-type: none"> • Word embeddings 	• Logistic regression	
(Przybyła & Teisseyre, 2015)	0.7489	N/A	<ul style="list-style-type: none"> • Polarity words, emotions 	• Distance-based approaches	
(Kocher, 2015)	0.7037	N/A	<ul style="list-style-type: none"> • The most 200 frequent terms 	• Distance-based approaches	

(Pervaz et al., 2015)	0.6379	N/A	<ul style="list-style-type: none"> • Punctuation marks • Sentence length • Question sentences 	<ul style="list-style-type: none"> • Distance-based approaches
(Kocher & Savoy, 2016)	0.2564	<ul style="list-style-type: none"> • Hashtags, RTs, URLs 	<ul style="list-style-type: none"> • Bag-of-words 	<ul style="list-style-type: none"> • Distance-based approaches
(Ucelay et al., 2016)	0.1538	<ul style="list-style-type: none"> • HTML and XML codes removal 	<ul style="list-style-type: none"> • Stylistic features with n-grams • 	<ul style="list-style-type: none"> • Distance-based approaches
(Kocher & Savoy, 2017)	0.465	N/A	<ul style="list-style-type: none"> • Top n terms by gain ratio 	<ul style="list-style-type: none"> • Distance-based algorithms
(Khan, 2017)	0.19	<ul style="list-style-type: none"> • HTML and XML codes removal 	<ul style="list-style-type: none"> • The most discriminant words per class from a list of 500 topic words 	<ul style="list-style-type: none"> • Distance-based algorithms
(Adame-Arcia et al., 2017)	0.1017	<ul style="list-style-type: none"> • URLs, mentions, hashtags removal • Contractions expansion 	<ul style="list-style-type: none"> • Bag-of-words • Emotion or/and language expression • Emotion, appraisal, admiration, positive/negative emotions, positive/negative words 	<ul style="list-style-type: none"> • Distance-based algorithms
(Don Kodiyan Florin Hardegger & Cieliebak, 2017)	0.6263	<ul style="list-style-type: none"> • URLs, mentions, hashtags removal 	<ul style="list-style-type: none"> • Word embeddings 	<ul style="list-style-type: none"> • Recurrent Neural Network •
(Miura et al., 2017)	0.6992	<ul style="list-style-type: none"> • URLs, mentions, hashtags removal • Lowercase words 	<ul style="list-style-type: none"> • Character and word embedding combination • 	<ul style="list-style-type: none"> • RNN and CNN with attention mechanism, max-pooling

					layer, fully-connected layer.
(Sierra et al., 2017)	0.6567	N/A	<ul style="list-style-type: none">• Word embeddings	<ul style="list-style-type: none">• Convolutional Neural Network	
(Schaetti, 2017)	0.615	<ul style="list-style-type: none">• URLs, mentions, hashtags removal	<ul style="list-style-type: none">• Punctuation signs removal• Number, out-of-alphabets removal• Character and word n-gram• TF-IDF combined with word embeddings and with beginning and ending 2-grams characters	<ul style="list-style-type: none">• Convolutional Neural Network	
(de Valkeneer & Mukhsinova, 2016)	0.3205	N/A	<ul style="list-style-type: none">• Bag-of-words	<ul style="list-style-type: none">• Class RBM	
(Pavan et al., 2013)	0.2843	<ul style="list-style-type: none">• HTML cleaning	<ul style="list-style-type: none">• Stylistic features (punctuation marks, capital letters, quotations, etc.)• Content features (Latent Semantic Analysis, bag of words, TF-IDF, dictionary-based words, topic-based words, entropy-based words, etc.)•	<ul style="list-style-type: none">• Maximum Entropy	
(Arroju et al., 2015)	0.6996	<ul style="list-style-type: none">• HTML code removal• Hashtags, URLs, mentions	<ul style="list-style-type: none">• Word n-grams• LIWC	<ul style="list-style-type: none">• Stochastic Gradient Descent (classification),	

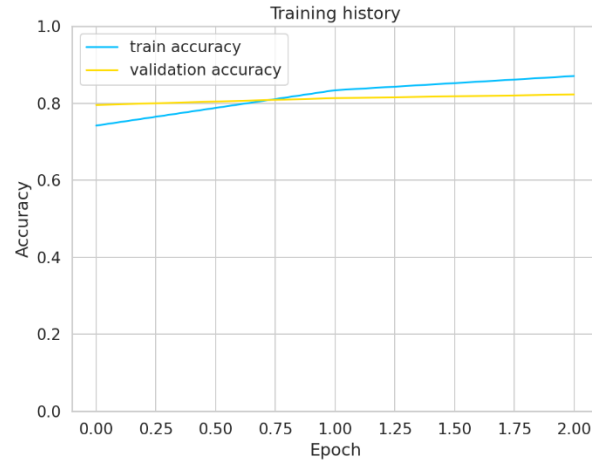
				Ensemble (regression)
(Şulea & Dichiu, 2015)	0.7378	N/A	<ul style="list-style-type: none"> • Verbosity • Character n-grams • TF-IDF n-grams 	<ul style="list-style-type: none"> • Ridge (regression)

APPENDIX C

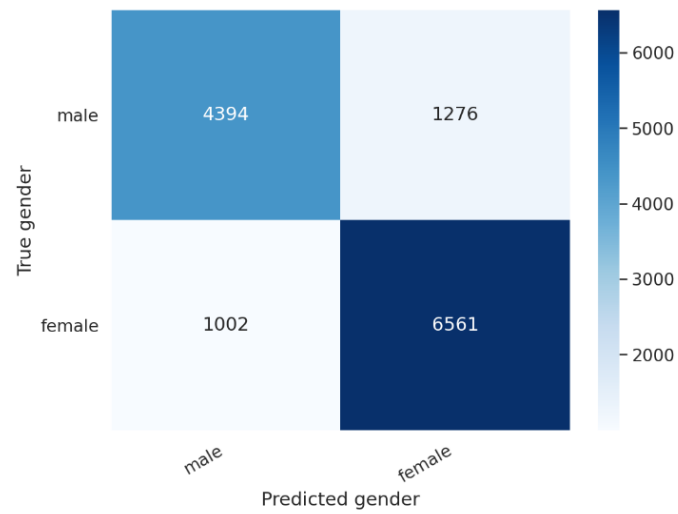
Experiments Charts, Results, And Confusion Matrices

The Experimentation of Text Preprocessing Effect On Gender Profiling

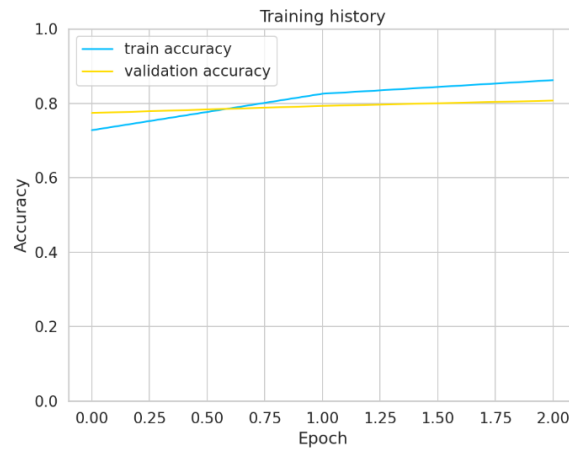
Case 1



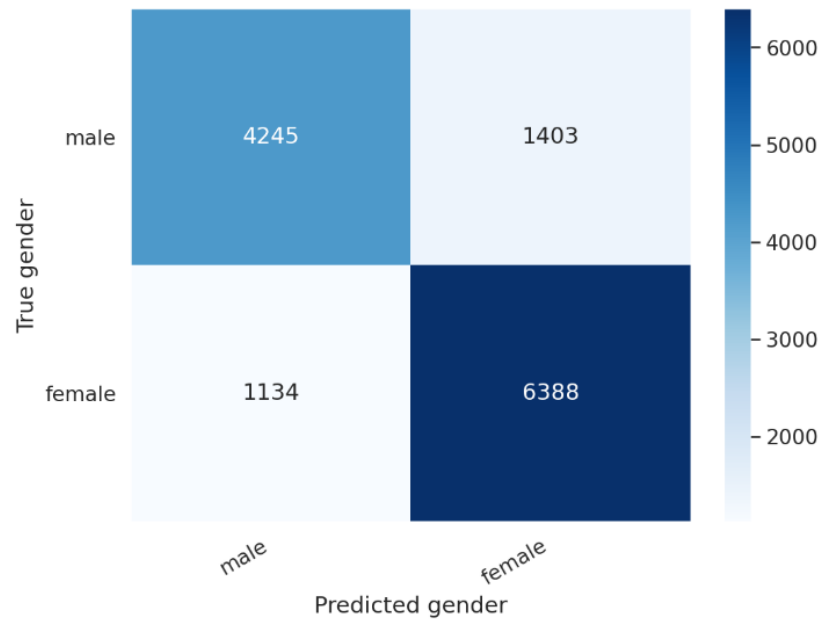
	precision	recall	f1-score	support
male	0.81	0.77	0.79	5670
female	0.84	0.87	0.85	7563
accuracy			0.83	13233
macro avg	0.83	0.82	0.82	13233
weighted avg	0.83	0.83	0.83	13233



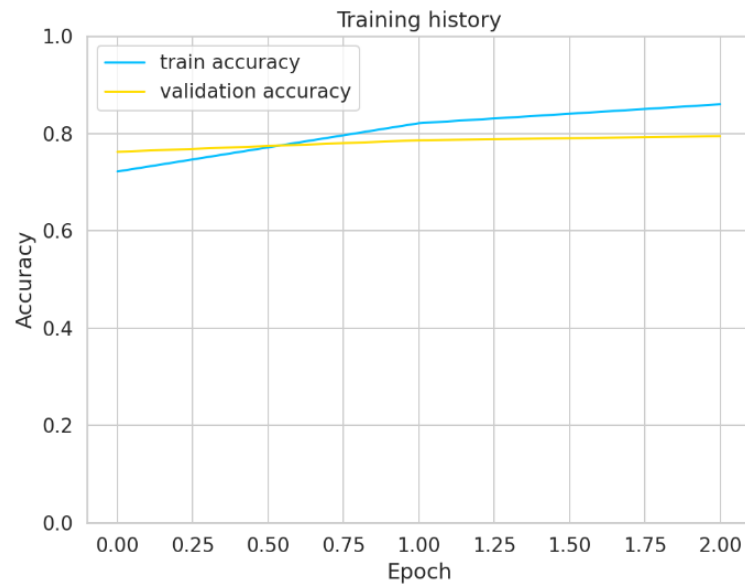
Case 2



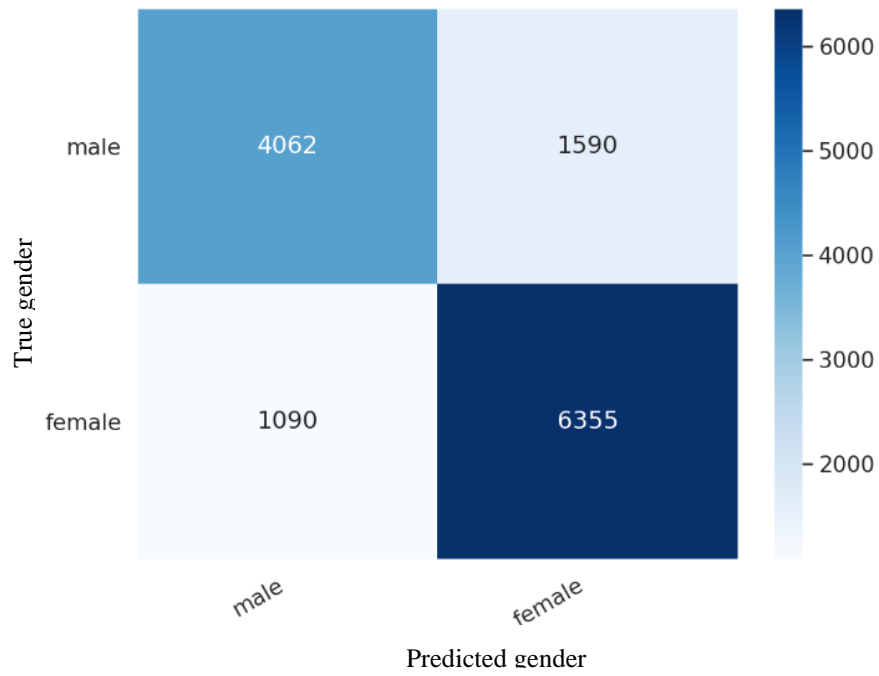
	precision	recall	f1-score	support
male	0.79	0.75	0.77	5648
female	0.82	0.85	0.83	7522
accuracy			0.81	13170
macro avg	0.80	0.80	0.80	13170
weighted avg	0.81	0.81	0.81	13170



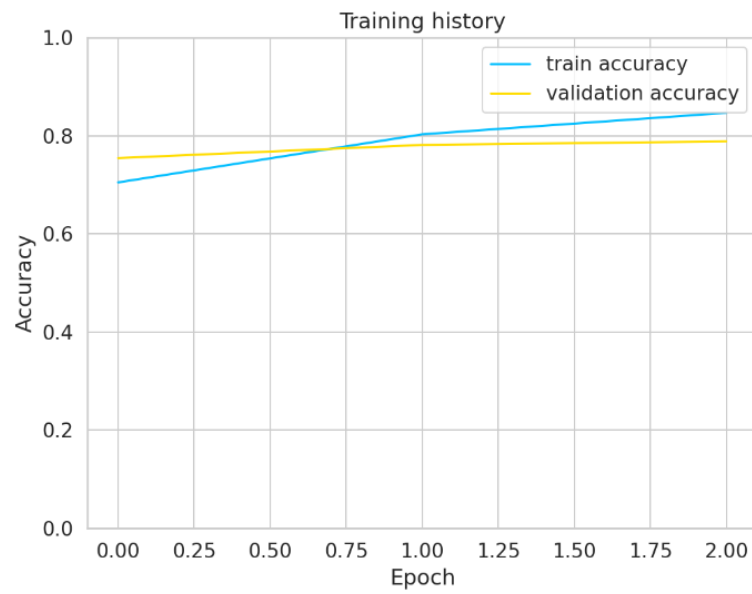
Case 3



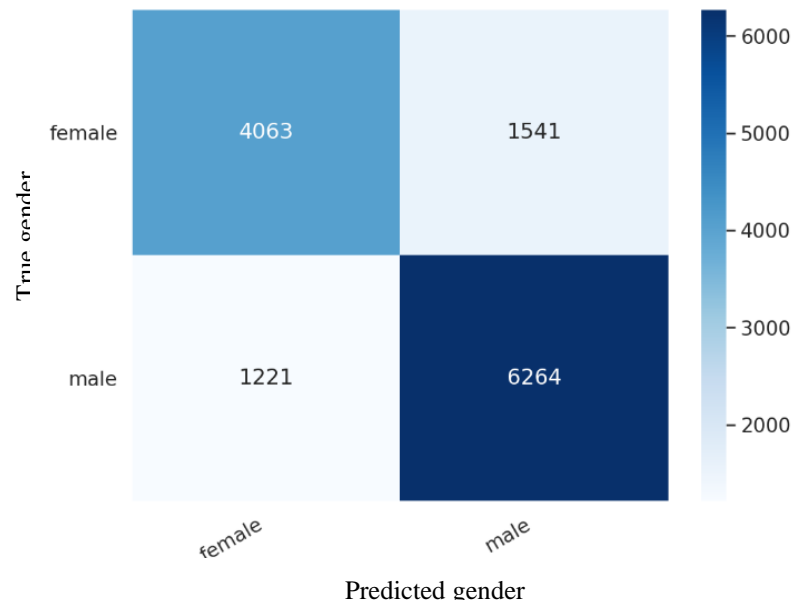
	precision	recall	f1-score	support
male	0.79	0.72	0.75	5652
female	0.80	0.85	0.83	7445
accuracy			0.80	13097
macro avg	0.79	0.79	0.79	13097
weighted avg	0.79	0.80	0.79	13097



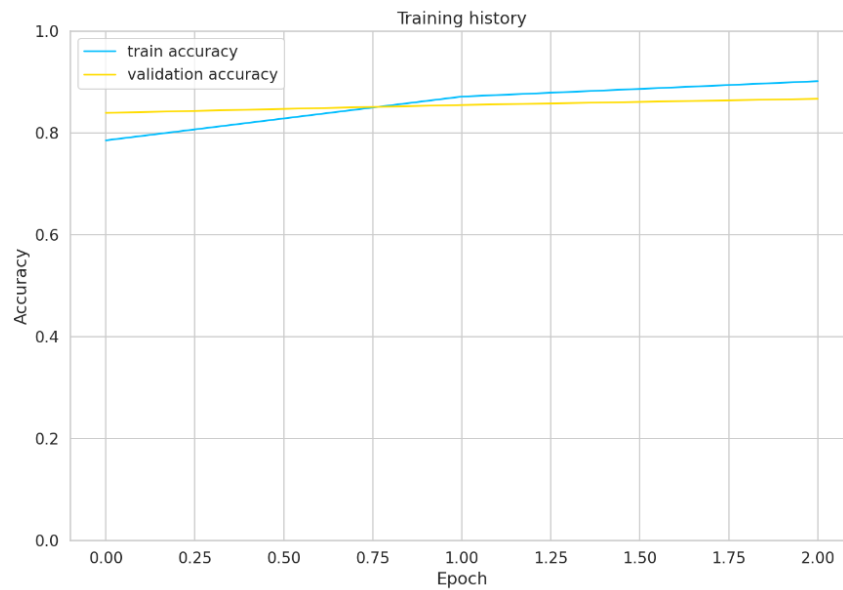
Case 4



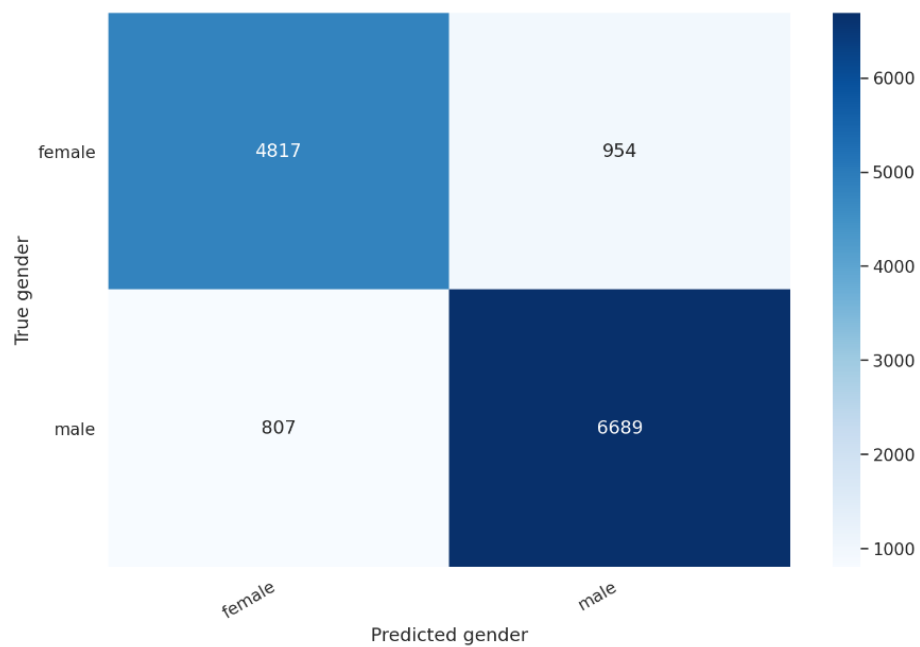
	precision	recall	f1-score	support
female	0.77	0.73	0.75	5604
male	0.80	0.84	0.82	7485
accuracy			0.79	13089
macro avg	0.79	0.78	0.78	13089
weighted avg	0.79	0.79	0.79	13089



Case 5



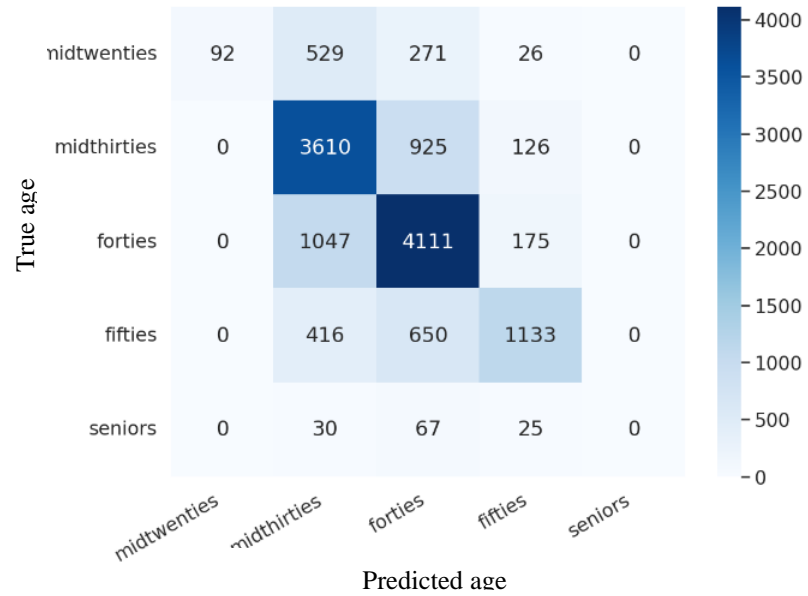
	precision	recall	f1-score	support
female	0.86	0.83	0.85	5771
male	0.88	0.89	0.88	7496
accuracy			0.87	13267
macro avg	0.87	0.86	0.86	13267
weighted avg	0.87	0.87	0.87	13267



The Experimentation of Text Pre-Processing Effect On Age Profiling

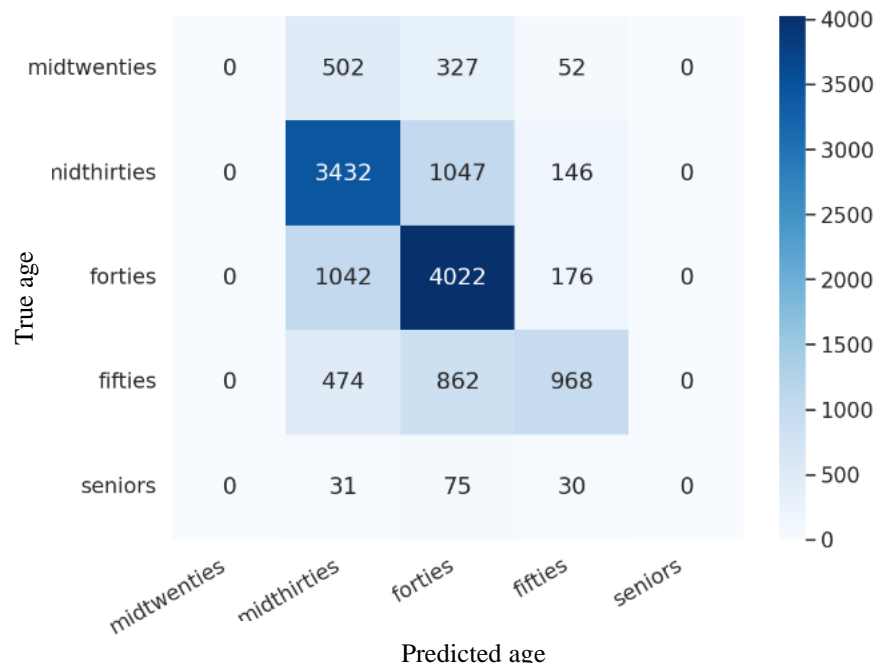
Case 1

	precision	recall	f1-score	support
midtwenties	1.00	0.10	0.18	918
midthirties	0.64	0.77	0.70	4661
forties	0.68	0.77	0.72	5333
fifties	0.76	0.52	0.62	2199
seniors	0.00	0.00	0.00	122
accuracy			0.68	13233
macro avg	0.62	0.43	0.44	13233
weighted avg	0.70	0.68	0.65	13233



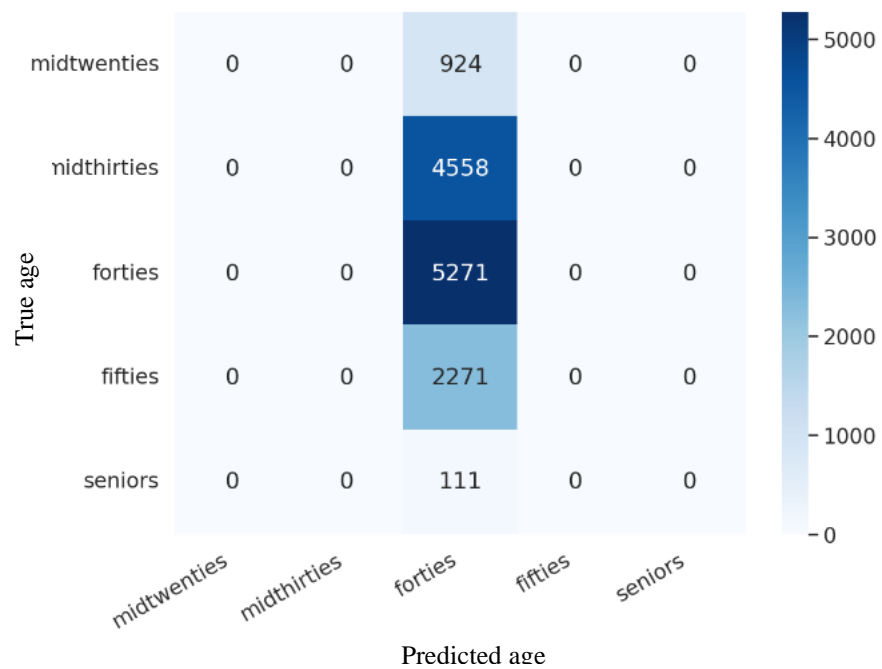
Case 2

	precision	recall	f1-score	support
midtwenties	0.00	0.00	0.00	881
midthirties	0.63	0.74	0.68	4625
forties	0.64	0.77	0.70	5240
fifties	0.71	0.42	0.53	2304
seniors	0.00	0.00	0.00	136
accuracy			0.64	13186
macro avg	0.39	0.39	0.38	13186
weighted avg	0.60	0.64	0.61	13186



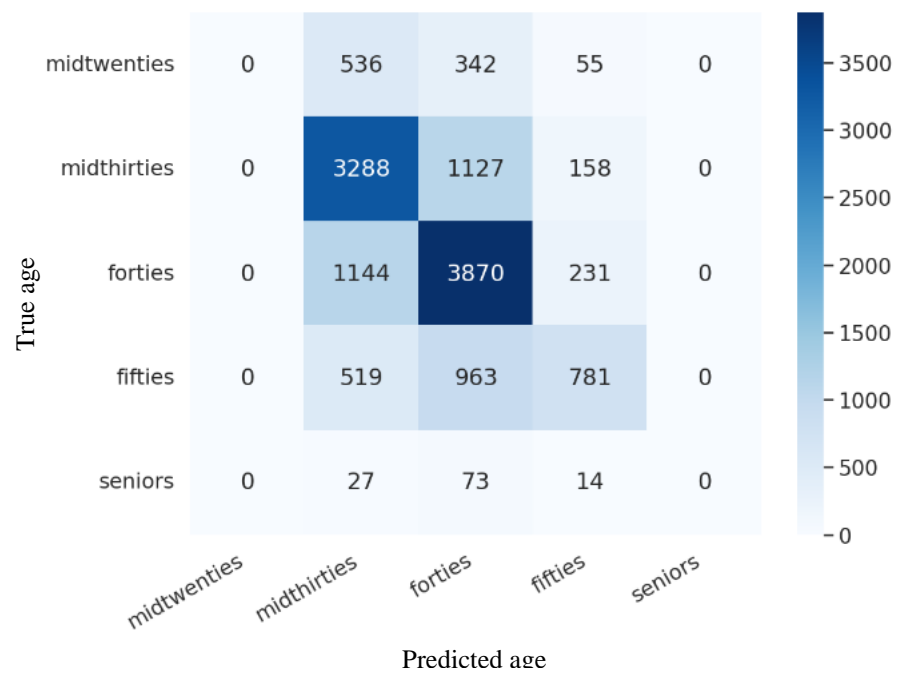
Case 4

	precision	recall	f1-score	support
midtwenties	0.00	0.00	0.00	924
midthirties	0.00	0.00	0.00	4558
forties	0.40	1.00	0.57	5271
fifties	0.00	0.00	0.00	2271
seniors	0.00	0.00	0.00	111
accuracy			0.40	13135
macro avg	0.08	0.20	0.11	13135
weighted avg	0.16	0.40	0.23	13135



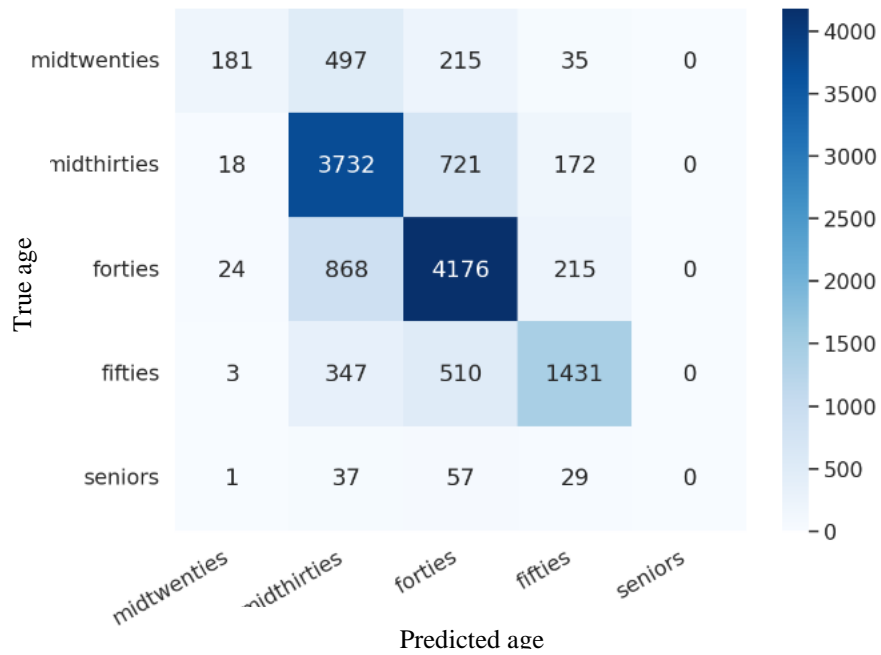
Case 4

	precision	recall	f1-score	support
midtwenties	0.00	0.00	0.00	933
midthirties	0.60	0.72	0.65	4573
forties	0.61	0.74	0.67	5245
fifties	0.63	0.35	0.45	2263
seniors	0.00	0.00	0.00	114
accuracy			0.60	13128
macro avg	0.37	0.36	0.35	13128
weighted avg	0.56	0.60	0.57	13128



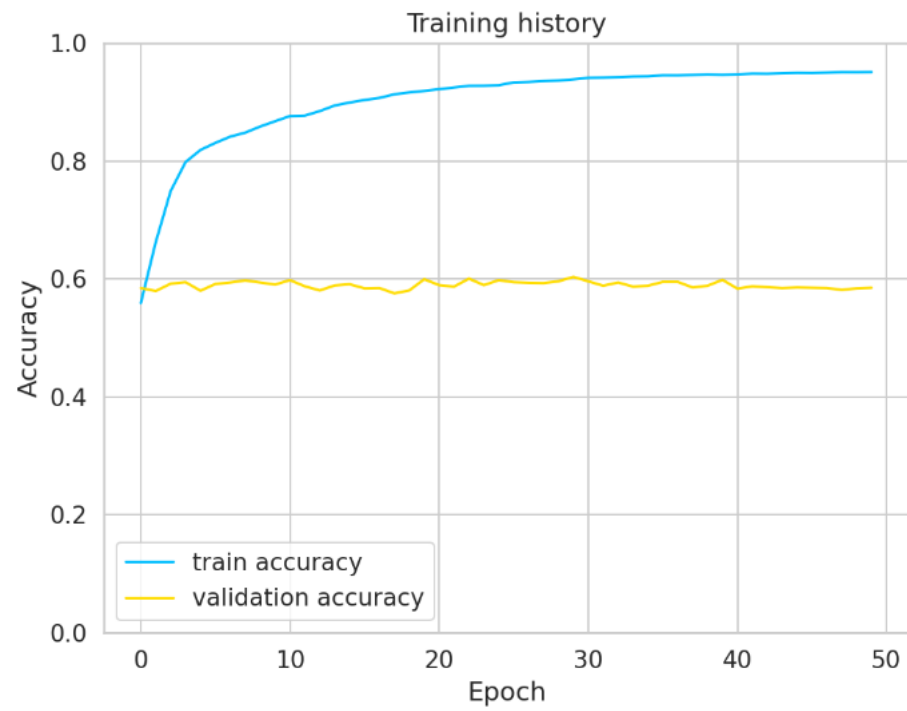
Case 5

	precision	recall	f1-score	support
midtwenties	0.80	0.20	0.31	928
midthirties	0.68	0.80	0.74	4643
forties	0.74	0.79	0.76	5283
fifties	0.76	0.62	0.69	2291
seniors	0.00	0.00	0.00	124
accuracy			0.72	13269
macro avg	0.59	0.48	0.50	13269
weighted avg	0.72	0.72	0.70	13269

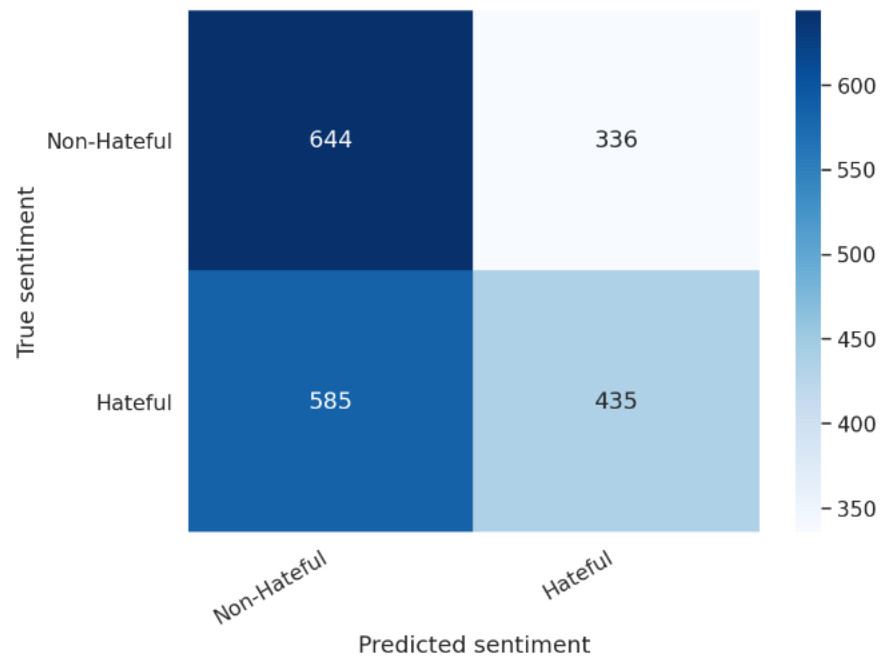


The case study of profiling fake news spreaders

English dataset profiling



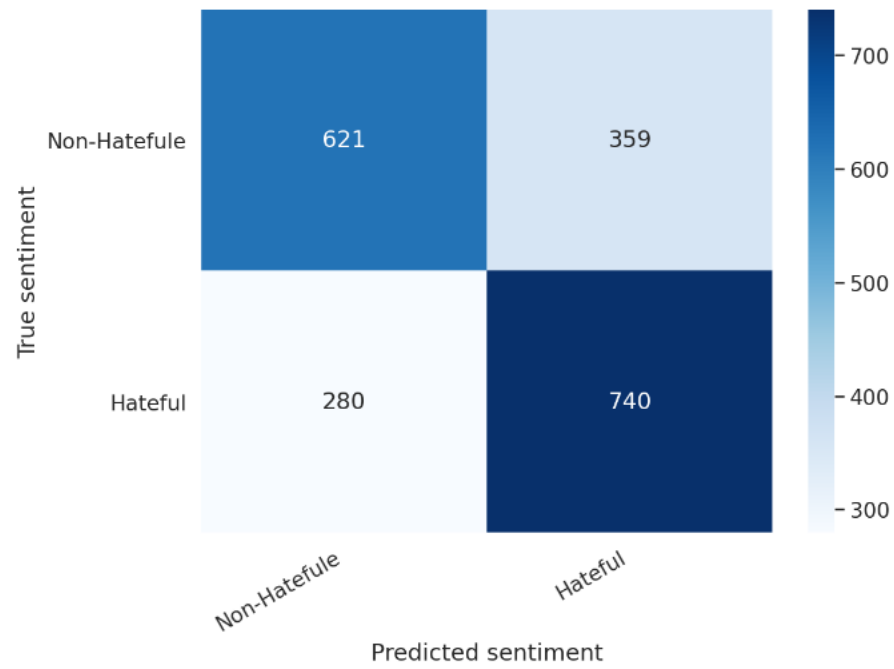
	precision	recall	f1-score	support
Non-Hateful	0.52	0.66	0.58	980
Hateful	0.56	0.43	0.49	1020
accuracy			0.54	2000
macro avg	0.54	0.54	0.53	2000
weighted avg	0.54	0.54	0.53	2000



Spanish Dataset





























































































































































	precision	recall	f1-score	support
Non-Hatefule	0.69	0.63	0.66	980
Hateful	0.67	0.73	0.70	1020
accuracy			0.68	2000
macro avg	0.68	0.68	0.68	2000
weighted avg	0.68	0.68	0.68	2000




















































































































































































































































































































































































































































APPENDIX D

The Frequency of Emojis In PAN 2017 Dataset

 11172 ' 8124	 717  713	 364  353	 225 🎵 221	 146  145	 97  96	 73 ♥ 72	 57  57	 48  48	 38 ⚡ 38
♥ 3759	 709	€ 340	 221	✓ 145	 95	 71	 57	 48	💎 38
 2907	 679	✳ 336	 209	 141	 95	 71	 55	→ 47	 38
 2617	 678	 332	 203	 141	 94	 70	 55	👤 47	 37
 2240	 674	 330	 202	 140	 92	 70	™ 54	 47	 37
 2237	 654	 327	 202	 140	 92	 69	 54	 47	 37
" 2112	 542	 316	 194	 138	 90	 69	 54	 46	♥ 37
" 2093	 541	 310	 190	 138	 89	☁ 69	 54	 46	 37
 1961	 521	 296	 189	 138	 88	 68	 53	 45	 36
 1716	 518	 295	 188	 136	 87	 67	 53	 45	 36
 1655	 514	 293	 186	 134	♂ 87	 64	 53	↑ 45	 36
 1295	 508	 292	 179	 130	 86	 64	 53	 44	 36
' 1096	 501	 290	 177	 124	♥ 86	🔗 64	 52	 44	 36
 1074	 492	 287	 176	 120	 86	♣ 63	 52	↓ 44	 35
 1055	 477	 287	 174	 118	 85	 62	 52	● 43	 35
— 1009	 459	 282	 173	 117	 84	 62	 51	 43	 35
— 990	 452	 271	 171	 116	 83	 62	 51	 43	 35

🙄 958	😄 438	😞 264	😟 171	☑ 115	🍷 83	🍷 61	👩 51	◯ 43	🌃 35
👩 927	429	👩 264	👉 166	🔍 113	🌸 82	📖 60	🍔 50	!! 42	📺 35
📖 920	👉 426	📖 261	🐶 165	😞 109	👥 82	🍇 60	🐱 50	🍁 42	🐎 35
😞 910	👉 423	✳ 255	🍹 156	🍷 107	🌴 79	👉 60	🌊 49	🎸 42	🔪 34
😞 907	😄 422	😄 246	✈ 154	🎤 107	👥 77	🐱 60	➔ 49	😞 42	🍋 34
🎉 854	• 421	♀ 246	☘ 152	🌈 105	👻 76	👉 59	🏀 49	⊕ 42	🐱 34
😊 750	😄 420	😄 233	👩 151	😞 102	75	🍴 58	★ 49	✕ 39	💣 34
😞 743	📖 420	🏀 229	🐱 146	😞 97	🏈 74	🐝 57	😞 48	🐘 38	📺 34
😞 741	❤ 387	❤ 227	😞 146	🦎 97	🍕 73	😞 57	🎥 48	🌞 38	🎬 33
🌻 33	👩 28	👉 22	🔪 19	! 16	🍏 13	🍌 11	🔑 9	👩 7	🏠 6
🐢 33	🎀 27	🌱 22	✂ 18	🍷 16	🐱 13	🦎 11	🔔 9	👩 7	🐱 6
🗽 32	🐟 27	↑ 22	🔑 18	🏠 16	🏠 13	🚫 11	👕 9	📺 7	🏛 6
🎵 32	🍏 27	📖 21	🍄 18	👉 16	🐮 13	🔍 10	📺 9	🎯 7	🔗 6
🚩 32	🌙 26	🎓 21	📺 18	👉 16	🍷 13	😞 10	🌂 9	🔪 7	🔑 6
🌍 31	💡 26	📖 21	👉 18	🐱 16	👩 13	🌊 10	🕒 8	▶ 7	🌞 6
🎯 31	⌚ 26	👩 21	🌱 18	🐱 16	👩 13	🕶 10	🏠 8	🎵 7	🔊 6
🐱 31	🐱 26	📺 21	🏠 18	🚫 16	👩 13	✚ 10	🏠 8	🔪 7	🌃 6
🍓 31	❤ 26	🔪 21	🏠 18	🍷 15	🎡 13	🐱 10	🐱 8	🐱 7	📺 6
🚶 31	🔪 26	👩 21	🐱 18	🐱 15	🐱 13	👩 10	' 8	🕒 7	🔪 6
🍩 30	🐱 25	👩 21	🐱 18	15	👩 13	🍷 10	🍌 8	▶ 7	📺 6
🍊 30	📺 25	👉 21	🔔 17	🐱 15	🍷 12	🍇 10	✚ 8	📺 7	❤ 6
— 30	👉 25	👉 20	🏠 17	🍇 15	👉 12	📺 10	👉 8	END 7	👉 6

	30		25		20		17		15		12		10		8		7		6
	30		25		20		17		15		12		10		8		6		6
	29		24		20		17		15		12		10		8		6		5
	29		24		20		17		15		12		10		8		6		5
	29		24		20		16		15		12		9		8		6		5
	29		24		20		16		14		12		9		8		6		5
	29		24		20		16		14		12		9		8		6		5
	29		23		19		16		14		12		9		8		6		5
	28		23		19		16		14		11		9		8		6		5
	28		23		19		16		14		11		9		8		6		5
	28		23		19		16		14		11		9		8		6		5
	28		23		19		16		13		11		9		7		6		5
	28		22	$=$	19		16		13	$\%$	11		9		7		6		5
	28		22		19		16		13		11	\times	9		7		6		5
	5		4		4		3		2	$\$$	2		2	\star	1	$-$	1		1
	5		4		4		3	$^{\circ}\text{C}$	2		2		2	\rightarrow	1		1		1
	5	$!?$	4		4		3		2	\angle	2		2		1		1		1
	5	\star	4		4	$\text{\textcancel{W}}$	3		2		2		2	\cdot	1		1		1
	5	\star	4		4	$\text{\textcancel{W}}$	3		2		2		2	\cdot	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1
	5		4		4	\cdot	3		2		2		2	\dagger	1		1		1

 5	 4	 3	 3	 2	 2	 1	 1	 1	 1	 1
 5	 4	 3	 3	 2	 2	 1	 1	 1	 1	 1
 5	 4	 3	 3	 2	 2	 1	 1	 1	 1	 1
 5	 4	 3	 3	 2	 2	 1	 1	 1	 1	 1
 5	 4	 3	 3	 2	 2	 1	 1	 1	 1	 1
 5	 4	 3	 3	 2	 2	 1	 1	 1	 1	 1
5	 4	 3	 3	 2	 2	1	 1	 1	 1	 1
 4	 4	 3	 3	 2	 2	 1	 1	 1	 1	 1
 4	 4	 3	 3	 2	 2	 1	 1	 1	 1	 1
 4	 4	 3	 3	2	 2	 1	1	 1	 1	
 4	 4	 3	 2	 2	 2	 1	 1	 1	 1	
 4	 4	 3	 2	 2	 2	 1	 1	 1	, 1	
 4	 4	 3	 2	- 2	 2	 1	 1	 1	- 1	