Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Endre Ulberg

"I can teach you to do that": On the analysis and prediction of sexual grooming and predatory behaviour against children in online chat forums

Master's thesis in Informatics
Supervisor: Björn Gamback
February 2022

NTNU
Norwegian University of
Science and Technology

Endre Ulberg

"I can teach you to do that'': On the analysis and prediction of sexual grooming and predatory behaviour against children in online chat forums

Master's thesis in Informatics
Supervisor: Björn Gamback
February 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Abstract

The advent of massive social media platforms has brought with it an increase in contact between children and strangers. It has become easier to exploit and harm children online due to large social media platforms having inadequate systems for supervision of interactions made on the platforms. Predators often isolate their targets from public spaces and move to private communication methods, making the grooming attempts hard to monitor. Increased access to young children online has made it a field of research to find grooming attempts in private chats with automated machine learning systems. The primary motivation has been improving the detection accuracy or detecting grooming earlier. The Thesis argues that neither of these issues is the most pressing one. Instead, the main issue within the field is the lack of access to sufficient amounts of high-quality data that are platform, iconography and time insensitive.

Using a quantitative approach to building algorithmic solutions, this Thesis explores the usage of features and algorithms for detecting predators. A framework was developed to test combinations of features and algorithms and leverage their strengths. To detect predators, six features and twenty-nine classical machine learning algorithms were trained on a preexisting dataset. The features were all lexical, meaning that the features are generated from text without any other data. The final solution explored Bag-of-words, Binary bag-of-words, Term Frequency, Term Frequency-Inverse Document Frequency (TF-IDF) , Linguistic Inquiry and Word Count (LIWC), and a combination of TF-IDF and LIWC. The algorithms were predominantly linear models, support vector machines, ensembles or variations on Naïve Bayes classifiers. The best combinations are aggregated to soft voting ensembles that combine the predictions from several algorithms. Lastly, the top-performing solutions are optimised with hyperparameter tuning.

The best performing solution gained third place in the current standings in detecting predators with a score of 0.947, measured by f0.5, the primary metric in the field. The solution was a two-stage approach using a Multi-layer Perceptron with TF-IDF to find suspicious conversations and a RidgeClassifierCV with TF-IDF to find which participants in the suspicious conversations were the predators. A high-performing solution shows that using a quantitative approach has merit as a framework for finding suitable solutions. The main contribution to the field of Sexual Predator Identification is the framework used to develop solutions and the proposed solution for the problem of detecting predators.

Sammendrag

I en tid der det blir flere store sosiale medieplattformer som aktivt brukes av unge, så økes interaksjonen mellom barn og fremmede. Det har blitt enklere å utnytte barn sin sårbarhet på nett gjennom digital plattformer. Manglende overvåkningstjenester på plattformene gjør at det kan være mange usette interaksjoner mellom brukere. Overgripere vil isolere barna så tidlig som mulig ved å flytte kommunikasjonen fra offentlige til private kommunikasjonsverktøy. Den økte tilgangen på barn gjennom internett har skapt et fagfelt som forsøker å finne overgripere i avlukkede chatterom ved hjelp av maskinlæring. Det har vært to primære drivkrefter i fagfeltet. Det ene er preventivt tiltak i form av tidlig deteksjon, der målet er å bruke så få interaksjoner mellom overgriper og barn som mulig på å avdekke overgreps forsøk. Det andre er å finne mer treffsikre løsninger.

Denne oppgaven vil argumentere for at det primære behovet i fagfeltet ikke er knyttet til disse to målene, men heller utfordringen knyttet til manglende data. Per i dag så er det ikke nok data som er uavhengig av plattform, ikonografi og tidsperiode. Dette gjør at det ikke er mulig å analysere de mer generelle mønstrene i overgrep på nett, og hvordan utviklingen er i forhold til tid, plattformer og nye former for skriftlige uttrykk. Ved bruk av en kvantitativ fremgangsmåte for å bygge algoritmer utforsker denne oppgaven bruken av egenskaper og algoritmer for å avdekke overgripere i tekst. Et rammeverk ble utviklet for å finne de beste løsningen og kombinere algoritmer og egenskaper for å nyttegjøre seg styrkene og svakhetene i de forskjellige løsningene. For å avdekke overgripere så ble seks egenskaper og tjueni algoritmer kombinert og trent på den mest brukte datamengden i fagfeltet. Den endelige løsningen brukte termfrekvens, termfrekvens inverse dokumentfrekvens (TF-IDF), bag-of-words, binær bag-of-words, Linguistic Inquiry and Word Count (LIWC) og en kombinasjon av TF-IDF and LIWC. Algoritmene i den endelige løsningen var fra flere familier, men de mest fremtredende var støttevektormaskiner, ensembler, lineære modeller og Naïve Bayes algoritmer. De beste kombinasjonene av egenskaper og algoritmer ble satt sammen i ensemblelæring. Optimering av hyperparameter ble brukt på ensemblene for å øke treffsikkerheten.

Den beste løsningen vil være den nåværende tredje beste løsningen i fagfeltet, med en score på 0,947 i f0.5, som er den primære metrikken for denne problemstillingen. Denne løsningen brukte en to-stegs arkitektur, der man først finner samtalene som er antatt å inneholde overgrepsmateriale, og etter dette avgjøre hvem i samtalen som var kilden til overgrepsinnholdet. For det første problemet så ble et nevralt nettverk med termfrekvens anvendt, og for å avgjøre hvem i samtalene som var overgriperen så ble det brukt en RidgeClassifier med termfrekvens. At løsningen fikk så høy resultat viser at kvantitative metoder for utvikling av løsninger kan gi verdifulle resultat. Den nye løsningen og rammeverket som har blitt brukt for å skape løsningen er de to primære bidragene fra denne oppgaven til fagfeltet.

Preface

The Thesis is part of my Master of Science (MSc) degree in Computer Science at the Department of Computer Science (IDI) at the Norwegian University of Science and Technology (NTNU) in Trondheim, Norway.

Firstly, I would like to thank my supervisor Björn Gambäck for allowing me to pursue a thesis within my chosen field, giving constructive feedback and thoughts, but more than anything being both patient and steadfast despite long breaks, the unusually long time it took to complete the Thesis.

Secondly, thanks must be extended to all researchers I have contacted during the Thesis. Thanks to April Edwards (ChatCoder2), Nikolaos Lykousas (Locard), Charles Ringer (TwitchChat) and Giacomo Inches (PAN-12) for access to several valuable datasets. Thank you to Patrick Bours for giving access to the completed features from the Social Behavioural Biometrics approach proposed in early 2021. For access to the linguistic inquiry and word count (LIWC), a rightful thank you to James W. Pennebaker. For providing information about the reason for the inaccessibility of the MovieStarPlanet (MSP) dataset, i would also like to thank Yun-Gyung Cheong.

Lastly, I would like to thank my roommate Aslak for encouraging and helping me to complete the Thesis and proofread the final drafts.

Endre Ulberg
Trondheim, 25th February 2022

Contents

List of Figures

List of Tables

1 Introduction

The rise of the Internet and social media has transformed how humans communicate and connect with each other. It has allowed people to connect without knowing anything about the other person. Open chat rooms like Twitch and Omegle make it possible to meet strangers. In some cases, personal information is exchanged which allows the conversations to move to more private and closed channels like Snapchat, SMS, or Facebook. When children are involved, the unsupervised nature of these more closed channels allows for the exploitation of said children. *Sexual grooming* is a process by which adults try to gain sexual favour from children or adolescents. The process will, in most cases, involve building strong bonds with the child, isolating them from their social network, and desensitising them from sexual advances. The ultimate goal for most grooming is control and free access to the child in question. Any person engaging in such behaviour is defined as a sexual predator or groomer.

The Thesis has as its primary goal detect predators. For this purpose, data that combines several sources containing predators, victims and regular Internet users are used to train and evaluate the machine learning algorithms. The high-performing solutions are analysed to detect patterns within the results that can be used to gain insights into promising features and algorithms for detecting predators.

This chapter gives a high-level overview of the background and motivation for the thesis. Following this, the research objective and research method are presented. Lastly, the contribution of this Thesis to the field and the structure of the Thesis are described.

1.1 Background and Motivation

Sexual grooming is sadly quite prevalent. As Girouard (2008) claims, one out of every seven children between the ages of 9 and 17 have been approached with sexual requests online. Following this, Kierkegaard (2008) found that up to 89% of approaches from groomers happen in chat rooms. With more and more children gaining access to the Internet as it becomes a public utility, several concerns are raised in regards to the health and well-being of all children. The ongoing coronavirus pandemic has significantly increased the activities related to Child Sexual Abuse (CSA) seen as an increase in both cases and key indicators like the use of Peer-to-Peer networks and activities of Child Sexual Abuse Material (CSAM) forums (Europol, 2020).

Online sexual solicitation has a significant impact on the life of the children involved. Victims of sexual abuse will in most cases experience feelings of shame, guilt and embarrassment (Baumgartner et al., 2010). Due to the extreme nature of sexual grooming, it is not uncommon for this to leave permanent damage or trauma symptoms. These

symptoms can manifest as a wide range of issues, including anxiety, depression, externalising problems, psychosomatic complaints, self-harm, PTSD and in extreme cases, suicide. Engagements with groomers often heighten the risk for further victimisation in the future, such as physical and sexual violence in their intimate relationships and entry into commercial sexual exploitation.

Motivated by a wish to ensure the safety of children online, The Conference and Labs of the Evaluation Forum (CLEF), which promotes research within natural language processing (NLP), holds yearly series of shared tasks. In 2012, a shared task was held for Sexual Predator Identification (SPI) with two problems presented:

1. *Identify the predators among all users in the different conversations*
2. *Identify the part (the lines) of the conversations which are the most distinctive of the predator behaviour*

The data published for the shared task has enabled researchers to analyse and predict the nature of online sexual grooming. Several approaches and techniques have successfully detected significant amounts of the predators in the datasets. However, few solutions attempt to use quantitative methods to find lightweight, accurate, and easy to implement solutions.

1.2 Goals and Research Questions

Goal *Determine if a quantitative approach to algorithm design performs well for Sexual Predator Identification*

Several researchers have presented high-performing solutions. However, most, if not all, have based their approach on qualitative information. This Thesis aims to use a quantitative method to make data-driven implementation choices to develop a solution to detect predators. The results from the developed solutions will be analysed to evaluate patterns for the features and algorithms to understand what impacts performance. The final purpose is to optimise a Sexual Predator Identification (SPI) solution.

Three research questions to support this goal are described in detail below.

Research question 1 *What existing approaches are the most promising for detecting predators?*

To better understand the state of the field, a literary review of studies related to the detection of grooming will be performed. The driving motivation in SPI is to improve machine learning approaches to detect predators. Finding and understanding which existing approaches are the most promising, in the sense of high performance in the primary evaluation metric f0.5, is a prerequisite to finding what can be considered promising to explore further and what can be considered baseline systems.

Research question 2 *What types of features are viable and optimal for use in detecting predators?*

Research Question 2 focuses on experimenting with different features to find which are the most impactful for detecting predators with the background gained from Research Question 1 to select which features should be explored as part of the experimentation.

Research question 3 *What categories of algorithms are most suited for use to detect predators?*

Research Question 3 aims to determine which of the implemented algorithms perform well across all architectures and feature sets to isolate the benefit of each algorithm.

1.3 Research Method

The Thesis applies an experimental approach to reach the goal of the Thesis. A literary review of the field was used to support the experimentation, gaining the required insight into the most promising approaches in the field. The review was performed as a Structured Literary Review combined with searching through references and citations from authoritative publications in the field to discover all the most relevant publications. A quantitative approach to experimentation was used with the promising solutions discovered as part of the literary review, combined with additions and adjustments to the approaches. The best solutions were determined by testing as many permutations as feasibly possible of features and algorithms and ranking them based on the primary metric f0.5. The experiments were performed by using data published as a part of the PAN-12 competition, which aimed to find the best solution to detecting predators.

1.4 Contributions

- A solution with an f0.5 of 0.947, gaining a third place in the current standings
- A quantitative framework for comparing similar solutions against each other
- A thorough literature review of the current field
- An analysis of the state of the field
- The invalidation of the results a recent paper, reporting the best performance to date with the use of Social Behaviour Biometrics (Wani et al., 2021).

1.5 Thesis Structure

1. **Introduction**
   The chapter gives an overview of the goals and motivations behind the thesis.
2. **Background Theory**
   Explains the relevant concepts and theories to introduce the prerequisites for assessing and understanding the thesis

3. **Related Work**
   Presents the literary search and previous work in the field of Sexual Predator Identification

4. **Data**
   Discusses the need for data in SPI and describes each dataset, the format, and the limitations of the datasets.

5. **Architecture**
   Describes the systems and features used to run experiments and analysis of the grooming data.

6. **Experiments and Results**
   Starts with the experimental setup, explaining the process and parameters used for analysis and detection of groomers. The latter half presents the results and findings.

7. **Evaluation and Discussion**
   Evaluates the experimental setup and analysis done and discusses the findings from those.

8. **Conclusion and Future work**
   Gives an in-depth explanation of the contributions, some closing remarks about the Thesis and some insights into what future work in the field can be.

2 Background Theory

In Sexual Predator Identification (SPI), there are several methods and techniques from Natural Language Processing (NLP) and Machine Learning (ML) that are prerequisites to understand how to solve the problem. This chapter will explain the necessary background for the Thesis. Firstly looking at how the data is preprocessed, followed by how we represent the texts to allow machine learning models to train on them. Then the machine learning techniques employed are described, followed by the core tools used.

2.1 Preprocessing

Preprocessing is the task of preparing data for further processing or analysis. In NLP, we commonly do two types of preprocessing called data preprocessing and text preprocessing. This section will first present some common ways of handling data preprocessing followed by techniques from text preprocessing.

2.1.1 Data Preprocessing

Given any data set for a task, there will be some data that is not needed or undesirable to complete a task. The saying "Garbage in, garbage out" is a fitting description of the goal of this step. Data filtering or data preprocessing is the task of determining what data is to be considered valid and valuable and what is to be view as errors, incomplete, noise or outliers.

Data preprocessing will, in most cases, be using metadata to determine what data points are desirable. For chat log data, an example could be that only messages that are in English can be utilised and therefore, all other data is discarded. Another example would be only keeping the messages containing more than ten words and chat logs with more than twenty messages, deeming the shorter ones too spare with information for us proper analysis to be performed.

2.1.2 Text Preprocessing

Natural language is ambiguous and free-flowing, meaning that the same meaning can be expressed in several almost similar ways or contain many words that do not add information to the text. Text preprocessing is used to transform a single text document to retain all the relevant information while removing all redundant information to increase the quality of the data.

Stop Words

Stop words are defined as the most common words in a given language. These words share the characteristic that they most often build syntax in a language but are not necessarily semantically significant. Stop word removal is removing these words that are considered bloat in the text, with the motivation of leaving only important data that allows for more accurate analysis for all the following techniques and models.

It is important to note that stop word removal can, in many cases, break syntactic analysis, so it will most often be applied in tandem with varying lexical analysis.

Lemmatisation

Natural language often contains words that have several forms or derivatives. The most common example being verbs like run, ran, running. Lemmatisation is the process of reducing every word to the root form, known as the lemma. Lemmatisation reduces the number of unique words while retaining the core meaning of each text, such that two sentences that talk about the same subject in different time forms will still be seen as very similar, allowing for more accurate comparisons between text.

Normalisation

Text normalisation is transforming text with the same meaning that is expressed differently to be represented the same way. Examples of text that should be normalised and which techniques we apply to them can be found in Table 2.1.

Name	Operation	Raw text	Normalised text
Same-casing	Uppercasing or lowercasing each word,	YES, I hate him	yes, i hate him
Emoji/Emoticon normalisation	Change groups of emojis to single emojis, or words that contain the same semantic content	:) :-) ;^)	smile smile smile
Noise Removal	Removing non-sensical charachters from text i.e.	nooooooo dont do. it	no dont do it
Word Expansion	Expand expressions to their original form	US don't mess with ya	The United States of America do not mess with you
Numeral conversion	Change numerals into numbers	23	twenty-three
Spell Correction	Fixing words that are spelled incorrectly	ambiougity	ambiguity
Synonym swap	Change all synonyms to their most common variation	retrowave	synthwave

Table 2.1: Normalisation techniques

Name	Text
D1	you are my little sister
D2	11 11
D3	77777 you are 77777

Table 2.2: Example documents

	11	77777	are	little	my	sister	you
D1	0	0	1	1	1	1	1
D2	2	0	0	0	0	0	0
D3	0	2	1	0	0	0	1

Table 2.3: Example documents transformed to a Bag-of-words representation

Lemmatisation, as explained in Section 2.1.2, is technically a type of normalisation but is considered a more extensive operation and, therefore, frequently categorised separately.

2.2 Text representations

This section is dedicated to explaining different formats for representing texts. Text representations are necessary because they allow computers to analyse and work with natural language more easily. Firstly the section will cover standard Natural Language Processing (NLP) techniques that have been applied. These techniques are used to transform natural language in different ways to allow for easier processing or analysis. Standard techniques used in NLP will be explained first, followed by information about other text representations specifically used for Sexual Predator Identification (SPI).

Table 2.2 is a modified subset of texts from the Twitch dataset described in Section 4.1.3. The corpus has been employed to explain the different text representations more precisely. For the rest of the chapter, A collection of texts is known as a corpus, while each text is known as a document.

2.2.1 Bag-of-words

A Bag-of-words (BOW) represents documents by using the frequency of each word in a document. The thought behind this is that the content of a document, represented by the occurrence of words and their frequency, is enough to create a valid representation of that document.

BOW is generated by first indexing every unique word in the corpus. This generates a vocabulary where the words are indexed in alphabetical order. Then, each document in the corpus is transformed into a vector with the count of each word in that specific document.

	11	77777	are	little	my	sister	you
D1	0	0	1	1	1	1	1
D2	1	0	0	0	0	0	0
D3	0	1	1	0	0	0	1

Table 2.4: Example documents transformed to a Binary bag-of-words representation

	11	77777	are	little	my	sister	you
D1	0	0	0.447	0.447	0.447	0.447	0.447
D2	1	0	0	0	0	0	0
D3	0	0.816	0.408	0	0	0	0.408

Table 2.5: Example documents transformed to a Term Frequency representation

2.2.2 Binary bag-of-words

The binary bag-of-words (Bin-BOW) is the same as the BOW representation in Section 2.2.1. The only difference is that the assumption is now that we only need to know the presence of words in a document to represent it. Meaning that it does not use frequency as regular BOW does.

Meaning that as in Table 2.3 the first message would be equal while the two last messages would have their 2's represented as 1's instead as seen in Table 2.4

2.2.3 Term Frequency

Term Frequency (TF) is a concept first described by Luhn (1958) as word frequency. A way to better perform queries in text, rather than looking for just if a document contained a term, it would also look at how many occurrences the words we were querying for had in that document, giving it the ability to rank the documents by significance for the query.

The pure Term Frequency is most commonly adjusted by taking the formula, which accounts for the total amount of words in the document, meaning that we get the how large percentage of the words in a document is that specific term. An example can be seen in Table 2.5

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

2.2.4 Term Frequency-Inverse Document Frequency

Building upon the work of Luhn, Spärck Jones (1972) extended the ranking algorithm. Term Frequency-Inverse Document Frequency (TF-IDF) also considers Inverse Document Frequency (IDF), meaning that it also employs how many documents the term appears in and the Term Frequency (TF) of that word in a single document.

	11	77777	are	little	my	sister	you
D1	0	0	0.373	0.490	0.490	0.490	0.373
D2	1	0	0	0	0	0	0
D3	0	0.880	0.334	0	0	0	0.334

Table 2.6: Example documents transformed to a Term Frequency-Inverse Document Frequency representation

The Inverse Document Frequency adjusts the ranking such that words that appear in few documents get a higher significance, which means that TF-IDF will rank words common in a document but scarce in the corpus highly for that document. Which is exactly what we can see as the difference between D2 and D3 in Table 2.6

$$IDF_j = \log \frac{n}{df_j}$$

$$TF - IDF = TF * IDF$$

2.2.5 N-grams

N-grams is a text representation that adds context to each word in a document. The representation is presented by having n consecutive words in a collection called an "n-gram", and the n denotes the number of consecutive words in each collection. The most common types of N-grams are called unigram (1), bigram (2), and trigram (3).

Below the explanation, a bigram representation of the D1 can be seen as an illustration. However, a note has to be made that there does not exist context words for the first and last word, so they will always have a NULL or empty value.

N-grams do not vectorise the words as the previous examples do, and it only adds more information to each word, meaning that n-grams can be used in conjunction with the previous representations. A note to make with N-grams is that they increase the amount of information and data in exchange for adding context, which increases processing time instead of the earlier representations, which are more information-dense than natural language.

$$[(NULL, you), (you, are), (are, my), (my, little), (little, sister), (sister, NULL]$$

2.2.6 MoodBook

Moodbook is an emotional lexicon that contains ten categories and words corresponding to each of these categories. The lexicon was built by Mudasir Ahmad Wani and Hussain (2018) as an extension of another emotional lexicon called EmoLex (Mohammad and Turney, 2010). Both of these built on the work of Plutchik (1980), which proposed the primary emotions, fear, anger, sadness, joy, surprise, disgust, trust and anticipation. In

	Fear	Anger	Sadness	Joy	Surprise	Disgust	Trust	Anticipation	EmotCat	Pos	Neg
D1	0	0	0	0	0	0	1	0	1	1	0
D2	0	0	0	0	0	0	0	0	0	0	0
D3	0	0	0	0	0	0	0	0	0	0	0

Table 2.7: Example documents transformed to a Moodbook representation

addition to the emotional categories, Moodbook has two sentiment polarities, positive and negative. The lexicon spans 1839 different words in total. Moodbook analyses the English language to get a picture of the person's emotional state while writing text.

As seen in the Table 2.7 only a single document had any emotional words. In this case, sister is a positive and trusting word. MoodBook is the first text representation thus far that depends on a pre-generated vocabulary, meaning that the size of the vectors is not dependent upon the text it tries to represent.

2.2.7 Linguistic Inquiry and Word Count

Linguistic Inquiry and Word Count (LIWC) is a lookup table developed by Pennebaker et al. (2015) that allows us to give a psycholinguistic profile to a text. The lookup contains the meaning of words in terms of social tendencies, behavioural patterns, personality, emotions, and some linguistic terms like Part of Speech tags, giving us a picture of a writer's social and psychological state while writing some document. This Thesis uses the 2015 version called LIWC-2015 that is composed of 6400 words, or root forms of words, together with a selection of emoticons. Each word is connected to a set of categories that describes the meaning of that word. In total, there are 74 categories that the words can be categorised into. An example is that the word "breast" or any variation "breasts" has the value categories ['bio', 'body', 'sexual']. For this reason, LIWC also generates quite lengthy variable-length text representations, meaning that it is not feasible to display the result of vectorising the example documents.

2.3 Algorithms

This section will cover the general concepts relating to the usage of machine learning algorithms that is relevant within SPI.

2.3.1 Supervised Learning

Supervised learning is a subset of machine learning (ML) within artificial intelligence (AI). In NLP, supervised learning aims to learn the relationships between some labels and some set of input data represented as vectors. An example is to decide if a male or a female wrote a text. Supervised learning is often described by using a student and a tutor. The student is given a task and tries to answer the task. The tutor will supervise the task, and if the task is solved correctly, the supervisor will give insights into why, and conversely, if the task is solved incorrectly, the tutor will explain some of the errors made

by the student. In Supervised learning, we use both data and labels, where the data can be a document that has been transformed with some text representation and a label that describes what that text is. The analogy is that the algorithm is the student, while the label is the answer sheet used by the teacher to grade the student. The algorithm is given a document and tries to predict the label. If the predicted label is correct, the algorithm will make it such that if a similar document is encountered, the new document will be assumed to have similar labels as the previous document. If the predicted label is not the same as the true label, an error function will try to some insight into what should be adjusted differently the next time a similar document occurs. When the algorithm has been trained on a sufficient amount of documents, which depends on the domain, the text representation, and the algorithm in question, the algorithm becomes a trained classifier that can predict the label of unseen documents. The data is split into training and test datasets to evaluate how well a supervised algorithm performs. The training set is used to learn how to predict by predicting and then be given the correct answers to adjust the predictions. The test set is only used to predict, and when a prediction has been made for all entries in the test set, it will be compared to the labels of the test set to generate an evaluation score.

Many issues can arise within the training or learning process for algorithms. Only the most prevalent one in SPI will be explained. This error is called overfitting, and as the name implies, the algorithm has been fitted too well to something. In the case of overfitting means that the algorithm has become too dependent on the training data and does not generalise well to other data sources. If an algorithm gets 100% accuracy during the training and gets a drastically lower accuracy when evaluated on unseen documents, it indicates a case of overfitting. This discrepancy between training accuracy and test accuracy indicates that the algorithm has become adept at recognising the different classes in the training set. There are many reasons this can occur, but a common one is that the algorithm is not learning anymore but simply memorising the documents and their labels. Another issue can be that the training set is very homogenous in some sense, and this issue can be compounded if the test set does not share the homogenous traits of the test set. The sources of these issues are numerous and outside the scope of the Thesis to provide an exhaustive list. Overfitting can be especially difficult in datasets where the training and the test data are drawn from the same source.

2.3.2 Ensembles

Ensembles are collections of classifiers that are used together to improve performance. There are several configurations of ensembles, some of them using classifiers connected in series and some in parallel. For this problem, only a specific variation of parallel configuration known as voting classifiers are considered.The core principle is that several supervised learning algorithms are trained on the same dataset, and then each of them is given a vote for what they think the label should be. There are two types of ways of assigning value to the votes from each subclassifier within a voting ensemble. The first is known as hard voting and is the most common. Each classifier gets a vote based on what class the classifier predicted the sample to be. The more advanced version is soft

voting classifiers that assign voting power by confidence for each specific entry.

In the same way that each classifier has a prediction, most classifiers also have confidence in their prediction. An example would be that the classifier is 60% certain that the first entry is a predator and therefore assign it to the class predator. Using the confidence allows classifiers to be more granular, and ensembles only sway the vote significantly if they are confident their prediction is correct. Both variations also have weighting schemes that are used to skew the importance of the different classifiers, allowing the voting ensemble to give more importance to the most robust predictors. In effect, both soft and hard voting allows for quite granular control, but the added value of using confidence allows it not to be uniform how important a classifier is but instead based on each specific prediction and the general importance of that classifier.

The goal is to allow each sub-classifier to mask its weaknesses and contribute with its strengths.

2.3.3 Hyperparameter tuning

Classifier and ensemble have an optimal configuration that will yield the best results. This optimal configuration can be found by using hyperparameter tuning. Machine learning models have two sets of parameters. The first set of parameters is static from the start of the training, and the second set is adjusted during training. The first set is the hyperparameters that decide how the second set of parameters is calculated during training. The second set of parameters is the parameters that get adjusted during the training to accommodate the correct and faulty guesses of the algorithms, meaning that the learning for the algorithm comes from adjusting this second set of parameters.

Since the hyperparameters define how the adjustments to the algorithms are made, they are very influential in how a classifier performs, and in many cases turning these hyperparameters well to the problem at hand can drastically improve the performance. There are several strategies for finding the optimal set of hyperparameters, and three of the most common ones will be explained in the following subsections.

Grid Search

The Grid search is the most intuitive and computationally heavy approach. Given a range of values for each hyperparameter, it will try all combinations in an exhaustive search, train each configuration and evaluate their performance. In effect, this is a very straightforward approach that is only feasible for small search spaces or in cases where the hyperparameters are already narrowed down to a smaller subset. The most redeeming quality of the strategy is that for any space, it is guaranteed to find the optimal solution, given enough time and processing power, and is a parallelisable problem since each tuning is independent.

Random Search

The random search is what the name implies. With ranges of values for each parameter, each parameter is assigned a random value from the respective range. As opposed to Grid Search, the only other difference is that Random Search has a finite amount of permutations that it should try. The advantage over Grid Search is that for fewer attempts, it is more likely that the tuning finds one of the promising regions of the search space, giving a better than average performance in a shorter time. It has most of the advantages except the guarantee of finding the optimal solution while having fewer downsides.

Baysian search

Bayesian strategies try to incrementally change some parameters to discover what nudges the performance in the right direction. So the more good performances are found within a search region, the more that search region will be explored further to find the optimal settings for the algorithm on this task. This is a combination of observing and understanding the relationships between parameters by isolating them when tuning them and searching for promising regions.

It is preferable to both Random and Grid search since it can give some information about the relationships between parameters and is faster since it explores the most promising regions instead of blindly or randomly testing combinations. However, it does sacrifice some of the ability to run tunings in parallel by having a partial dependency on earlier results.

2.4 Evaluation Metrics

Assessing the performance of machine learning algorithms can be done with several metrics. The section will present the metrics used to evaluate algorithms in SPI.

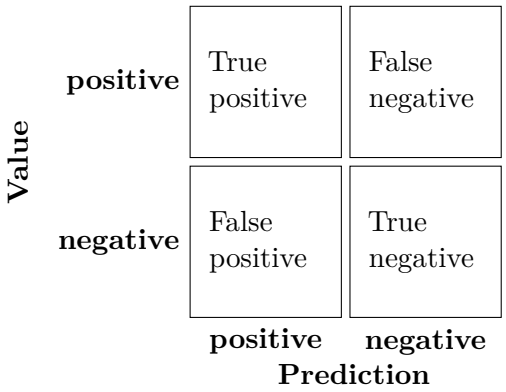


Figure 2.1: Binary Confusion Matrix

Figure 2.1 shows the relation between a prediction and the truth. There are two correct guesses, the true positive (TP) and the true negative (TN), and two incorrect guesses. A false positive (FP), also known as a Type I error, denotes that something is true when it is not. In the case of Sexual Predator Identification (SPI), this would be a case where a system predicts a non-groomer to be a groomer. Conversely, if a predatory is falsely believed to be innocent, this would be a false negative (FN), otherwise known as a Type II error. The two true cases are when a predator or non-groomer are correctly predicted. In Sexual Predator Identification, a subfield of information retrieval, it will be correct to use the terms false positive and false negatives, so for the remainder of the Thesis, that will be the terminology used.

Each metric we use to assess performance in algorithm solutions is based on the occurrence of each of these four values. The most well-known of the metrics is called accuracy and can be defined as

$$\frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy is concerned with how many per cent of all predictions were correct. Accuracy is, in many ways, a very coarse evaluation metric that is a good starting point and quite efficient in cases where there is a balance between the classes.

The three other metrics are recall, precision and f-score, and share an attribute that accuracy does not. The metrics are all measured from the perspective of a single class, meaning that each will have a different value depending upon assessing the performance for identifying predators or victims with the same algorithm. We can easily verify this in the formula for the metrics, where accuracy is the only metric to use TN and FP.

For the rest of the explanation, we will look at the scoring metrics from the perspective of predators.

$$\frac{TP}{TP + FP}$$

Precision will, in this case, ask, when the prediction is a predator, what is the certainty that the prediction is correct. If no victim is ever predicted to be a predator, the precision will be 100, or in other words, no prediction has been a false positive.

$$\frac{TP}{TP + FN}$$

What recall tells us is how many per cent of all predators were classified as predators, and it can be rephrased as "how many predators got away?" If no predator is ever predicted to be a victim, the recall will be 100. Or, in other words, the absence of false negatives.

$$(1 + \beta^2) * \frac{precision * recall}{(\beta^2 * precision) + recall}$$

F-score is a metric used to evaluate a classification system based on both recall and precision, using a harmonic mean function between the two. F-score generally refers to

the F1-score, which is the equally weighted version. However, adjusting the β in the function will skew the f-score towards either precision or recall. In the cases where the F-score is not an F1-score but has been skewed, it is also known as an F-beta score. Setting the beta higher will favour recall while setting the beta lower will favour precision. The important thing with F-score is that it is a measure that considers both precision and recall, making it ideal as a balanced measure for finding the performance of a model for a single class.

Cross-Validation is a model validation technique used to assess the performance of a model by splitting the training dataset into equal pieces and training on some, and validating on other parts of the training set.

There are several applications where this approach is favourable. Firstly it avoids outliers in training results by also getting training results that have been assessed on unseen data. A second application is for hyperparameter tuning, where unbiased evaluations during training are important to decide what search regions are valuable. Another common reason to use it is that it helps uncover errors in learning by using the validation results to compare against the test results. If they are comparable, the solution usually has few errors, but significant disparities point towards errors in the learning, which in many cases comes in the form of overfitting.

The training set can be segmented into two sets to avoid this issue, one for training and one for validation. However, splitting the training set, which comes from a single source, introduces a new problem. How can it sample that dataset correctly to have valid results when testing with the validation set. Controlled randomisation would help by having the same balance of classes between the training set and the validation set. The controlled randomisation is called a stratified approach. However, splitting the data in two decreases the size of the training set drastically, which again is not beneficial since quality models are dependent on as much data as possible up to a certain point.

An improvement is K-fold cross-validation, which is performed by first splitting the training data into equal size folds. Then the algorithm is trained with one of the folds as the validation set, while all the rest are used training data. This procedure is repeated using each fold as a validation set once, training the algorithm K times. The validation scores from the K training are averaged, giving the final validation score. K-fold cross-validation allows the whole dataset to be trained on, and the whole training set has been used as a validation set.

Both the evaluation of which algorithms are most appropriate to use for ensembles and how the in-training evaluation of the hyperparameter tuning has been done with a stratified ten-fold cross-validation.

2.5 Tools

Pandas Pandas is a data handling tool for Python made to handle large amounts of data and efficiently manipulate it. It operates using rows and columns of data and provides optimisation and support for many data formats.

The data frames it utilises is often interoperable with machine learning tools, making it a natural way to handle data.

Scikit-learn Scikit-learn (Sklearn) is a package containing many of the most used methods in machine learning. It is a toolkit with preprocessing, feature extraction, algorithms and evaluation functions implemented.

LazyPredict LazyPredict is a package building on top of Sklearn that allows for training many classifiers in series quickly. It implements an interface for training all classifiers from Sklearn, XGBoost, and LightBoostG on the same data and returns evaluations of the training and trained classifiers.

Modifications to LazyPredict To accommodate the needs of this thesis, LazyPredict was modified to remove the automatic transformation of features. LazyPredict will, by default, apply transformations to numeric and categorically data and discard any other data. To allow for better integration with textual data, LazyPredict was rewritten to skip the automatic transformation.

LazyPredict also avoids training some of the classifiers from Sklearn because they are not compatible with all tasks. These restrictions were modified to allow for the use of more classifiers.

Lastly, LazyPredict does not support model selection and does not report enough data to find overfitting. An extra suit was added to the package allowing for custom scoring metrics, and implementing cross-validation, such that the tool could be used for model selection and report all the information needed for experimentation and results.

NLTK Natural Language ToolKit is a toolkit that aids in the processing of text and the generation of features. It is commonly used for most text operations and feature extractions due to having all the standard operations for NLP and being properly optimised. It also implements several machine learning algorithms and more advanced features like Named Entity Recognition and sentiment analysis.

Optuna Optional is self-described as an automatic hyperparameter optimisation software framework. In essence, it is a full suite of hyperparameter tuning algorithms that correctly log all events during the training to give proper insight into the hyperparameter tuning. It is built to have bayesian search as the default and can visualise the results of training. The framework has been popularised together with LazyPredict on Kaggle, a website for Machine Learning competitions.

3 Related Work

Related work will present a review of research into grooming analysis and prediction. The review encompasses both statistical analysis to the extent needed for grooming analysis and the machine learning approaches used to predict grooming. Following this, state of the art based on stated results will be presented.

3.1 Topology and characteristics of the field

Due to the somewhat unique nature of Sexual Predator Identification (SPI), being a small and cross-disciplinary field of study, the research in the field also has some less usual aspects that should be discussed to give a proper understanding of the reasoning behind the structure of the chapter. This section is dedicated to describing the most prevalent of these unique features and how they affect a literary review.

The first, and probably most intuitive feature, is that researchers with different backgrounds will name their publications according to their field of study. An important point within this is that researchers from different fields have different contributions to SPI. While psychology gives the basis in theoretical frameworks that can be used to model or analyse predators, informatics often has practical implementations of these theoretical frameworks. At the same time, criminology aids in studying the prevalence and how predatory behaviour is conducted. Publications from each profession are essential and should be explored to gain proper insight into the problem domain. An example of how many ways one can word problems in this domain, many publications use the words paedophilia, grooming or online predatory behaviour interchangeably. In addition, online can often be substituted for cyber, as in cyber predatory behaviour.

The second and probably more hidden feature of the field is that several clusters of researchers work in SPI. These clusters are often made up of a leading researcher with an older, widely cited publication and then new publications from that leading author or the co-authors of the original publication in the field. Most of these clusters keep looking further into the issue presented in the first paper, meaning that they specialise in a single issue. The point of interest with these clusters is that even though several of the newer publications often are of high quality, and with improved methodologies, architectures, analysis or even an extension allowing for broader application, they rarely come with entirely new information, meaning that the original publication covers the most important insights, within the cluster. Since this means that a single publication from one researcher can spawn up to five or more new publications inside the same family, looking at the same problem, it is hard to find which of the six is favoured to be cited by newer papers. In effect, this makes it so that there are complex and sparse networks of

publications in the field.

The two issues mentioned above are both compounded by the last issue, which is hard to pinpoint, but some publications containing interesting results in SPI are "hidden" because they have names that are entirely hidden when searching for queries related to the detection of the sexualisation of children.

Between these three issues, neither a structured literary review, using keywords to generate queries, nor an approach by selecting good publications to search for new publications will give sufficiently good results on their own. The most appropriate way to untangle the connections between publications is a hybrid approach, using both SLR to find the first publications and then search in the references and citations from each of these publications and traversing as far as is deemed fruitful.

Another characteristic of the field is that it is not connected to the topology, which should be discussed. Within the machine learning part of the field, there are several axes by which one can examine the field. A single paper will often have a goal, an architecture or ML model of choice, a set of features they use, and in some cases, the publication is built upon some theoretical framework.

Most publications share at most three of these aspects with each other, and often only two aspects or only a single. In addition, the field does not move uniformly over time. Often a subfield like early detection of predatory chats will move in a burst over a few years and then be more dormant for a period. These two events make it such that structuring the related work by time will generate recurring themes, and structuring by a feature will make the same papers mentioned with different contexts in different sections. There is no perfect common ground that allows for a structured explanation of the field that encompasses both the depth and the breadth of the field.

The closest compromise that was found was to explain each publication's most exciting part instead and order them in a hybrid format, starting chronologically and moving to a split between goal and features.

3.2 Literary Review

A literary review was conducted to discover relevant solutions and possibilities in the field. The goal of the review is to structure how to search for publications and the processing of information in the publications. Firstly, the section explains how the Structured Literary Review (SLR) was completed, followed by the SLR's shortcomings, mainly due to the cross-disciplinary nature of the field as described in Section 3.1. Lastly, the section describes the other methods employed to search for literature used in combination with SLR to improve precision and recall in the search.

3.2.1 Structured Literary Review

A Structured Literary Review (SLR) is a process to find and review literature. The core of the process is defining search terms, using them on different search engines and then use inclusion and exclusion metrics to discard or evaluate different papers.

Group	Term
1	Child
2	sex, sexual
3	abuse, grooming
4	predator, pedophile
5	classification, detection, analysis

Table 3.1: Terms used for Structured Literary Review

Queries
1, 2, 3
1, 2, 4
1, 3
2, 3
2, 4

Table 3.2: Query configurations for Structured Literary Review

Defining the search terms is a search strategy, while the evaluation metrics are called a review protocol.

Search Strategy

Targeting the correct field of study and narrowing the search scope are core parts of the search strategy. To ensure that the search strategy aligns with the research questions, we define groups of terms from the field and combine these groups to generate specific queries to input to search engines. In addition, it is correct to add some boundaries to the search.

The groups and terms used will differ based on the field of study, so some prior knowledge of the field is required to pick appropriate terms and group them correctly. Finding the standard terms used in the titles and keywords is possible with some relevant articles, giving a baseline for search terms. To generate a group, we treat each term as a synonym, using the logical OR operator to generate a group. Queries are several groups used in conjunction, meaning that we use a logical AND operator between some groups to generate a single query.

Table 3.1 and Table 3.2 shows the terms and queries used for the SLR for this Thesis.

An example of a 2,4,5 query would be : ("sex" OR "sexual") AND ("predator" OR "pedophile") AND (“classification” OR ”detection” OR ”analysis”)

To further narrow the scope of the search, two additional parameters were added. Firstly only the first five pages of results from any given search engine would be considered. The rapid degradation in relevance for the search results required some cut-off points to avoid spending too much time filtering through irrelevant search results. In addition to the page limit, a time limit was added. No papers written before 2007 were considered

Primary Source
ResearchGate
SemanticScholar
Secondary Source
Arxiv
Science Direct

Table 3.3: Search Engines used for Structured Literary Review

due to the field's significant increase in activity due to the 2012 CLEF conference.

The publications were retrieved from several databases and search engines. The search engines were grouped into primary and secondary sources due to the varying scope and size of the databases. Table 3.3 shows that two search engines were chosen as the primary sources to conduct SLR, while two secondary sources were considered or briefly cross-referenced against to look for significant discrepancies in the findings from the primary sources.

Review protocol

The papers were evaluated against several metrics. They were evaluated using the inclusion and exclusion criteria (ICs/ECs) seen in Table 3.4 to decide which papers should be investigated further. Then, a set of quality assessment metrics were employed to score each paper by quality and relevance.

Due to having many inclusion criteria, they were also split into primary and secondary criteria. Studies were assessed based on the abstract, title, and background using the primary inclusion criteria. If the study passed two out of three, it would be investigated further.

ECs are used as a counter-part to ICs, as a tool to verify if a paper's investigation should be concluded. If either of the ECs were true, the study was outside the scope of the Thesis. Some papers could be allowed further investigation based on secondary ICs, or if the publication seemed to have some value that eluded the IC.

Primary Inclusion Criteria

The main concern of the study is child sexual abuse problems related to the use of the Internet.
The study is a primary study presenting empirical results.
The study focuses on detection of predators in chatrooms

Secondary Inclusion Criteria

The study provides a solution or algorithm to solve the problem
The study uses a good dataset

Exclusion Criteria

The paper is purely about psychology
The study is a small scale analysis that cannot be automated

Quality Criteria

There is a clear aim of the research
The study is closely related to other studies in the field, either by good references, or by being referenced by other papers.
Is the study novel?
Are the results, discussion or the findings significant?

Table 3.4: Criteria for Structured Literary Review

Name of publication	Author	Source
The communicative modus operandi of online child sexual groomers: Recurring patterns in their language use	Lorenzo-Dus et al.	Research Gate - Q1
Mitigating Online Sexual Grooming Cybercrime on Social Media Using Machine Learning: A Desktop Survey	C.H. Ngejane	Research Gate - Q1
Linguistic analysis of chat transcripts from child predator undercover sex stings	Drouin et al.	Research Gate - Q2
Characterising Pedophile Conversations on the internet using Online Grooming	Gupta et al.	Research Gate - Q2
Chat Analysis Triage Tool: Differentiating contact-driven vs. fantasy-driven child sex offenders	Seigfried-Spellar et al.	Research Gate - Q2
Ensemble Method for Sexual Predators Identification in Online Chats	Fauzi et al.	Research Gate - Q3
Sexual predator detection in chats with chained classifiers	Escalante et al.	Research Gate - Q3
Toward Spotting the Pedophile Telling victim from predator in text chats	Nick Pendar	Research Gate - Q3
Detecting Sexual Predators in Chats using Behavioural Features and Imbalanced Learning	Claudia Cardei	Research Gate - Q3
Detection of child exploiting chats from a mixed chat dataset as a text classification task	Rahman-Miah et al.	Research Gate - Q3
An Intelligent Online Grooming Detection System Using AI Technologies	Anderson et al.	Semantic Scholar - Q1
C3 -Sex: A Conversational Agent to Detect Online Sex Offenders	Rodríguez et al.	Semantic Scholar - Q1
Detection of cyber grooming during an online conversation	Halvor Kulsrud	Semantic Scholar - Q1
Sexual Predator Identification using Machine Learning on Android	Matthias Vogt	Semantic Scholar - Q2
WEBMINING AGAINST PEDOPHILIA	Grzegorz Filcek	Semantic Scholar - Q2
A survey on text mining in social networks	Rizwana Irfan et al.	Semantic Scholar - Q3 - Snowballed
A Two-step Approach for Effective Detection of Misbehaving Users in Chats	Villantoro-Tello et al.	Semantic Scholar - Q2 - Snowballed
Conversation Level Constraints on Pedophile Detection in Chat Rooms	Claudia Peersman et al.	Semantic Scholar - Q2 - Snowballed
Technical Mapping of the Grooming Anatomy Using Machine Learning Paradigms: An Information Security Approach	PATRICIO ZAMBRANO et al	Semantic Scholar - Q2 - Snowballed

Table 3.5: Inital results from Structured Literary Review

Evaluation of the SLR

SPI is a cross-disciplinary field, so many articles or publications can match all the search queries but focus differently. The first set of results which as seen in Table 3.5 were discarded very early on were primarily focused on psychology. Even after evaluating and updating the terms, groups, and queries to reflect the technical aspects of the Thesis, the queries still predominantly returned irrelevant results. The issues stemming from the results are two-fold, firstly we have low precision, and secondly, the recall is low. Such issues point to the queries being both too broad and with the wrong focus. Such indications should be taken as a prompt to reassess the baseline of the SLR, possibly introducing some way of balancing the terms used in each group to fit the focus better.

The underlying reasons for the low precision and recall are mostly likely the topology and characteristics of the field as described in Section 3.1

The discouraging results from the SLR initiated secondary strategies for finding relevant articles.

3.2.2 Snowballing and reverse searching

In addition to SLR, two other methods can be employed to find relevant publications. Snowballing consists of viewing which works an article references. Reverse searching looks at which new articles cite an older relevant article.

Snowballing aims to discover related publications to ensure a complete overview of a section of the field. Snowballing can be viewed as a modified depth-first search (DFS), with ECs/ICs to prune some nodes. Fitting the description of a DFS, it is important to decide a depth one is willing to snowball. Setting a max depth helps focus the search and ensures a more uniform process. Snowballing turned out to be one of the most efficient and consistent tools for finding relevant publications for SPI. Note that this points towards most newer publications being related to at least one authoritative older article.

Reverse searching is performed by finding the most autorotative articles in a field and looking at which articles have referenced them. This strategy allows us to find the newest addition to the field through association. For the best results, reverse searching with a previous state of the art publication will find every paper that has been compared to this or benchmarked their results against the previous state of the art.

These two methods work best in conjunction, searching both forwards and backwards from articles of interest. Such a search strategy is not a genuinely structured process, as the search trees can get quite large and have complex routes to trace. However, the benefit of considerably higher precision and recall was a necessary trade-off.

3.3 Sexual Predator Identification

Sexual predator identification is divided into several sub-disciplines. There are various ways to detect deviant behaviour like age disparity, Luring Communication Theory, psycholinguistic profiling, to name a few. This section is structured to accommodate

the nature of the field, firstly presenting the early work, pre-dating the highly-influential PAN-12 dataset, followed by a competition held where the dataset was presented, lastly ending with a subsection for each sub-discipline of interest.

3.3.1 Early work

In psychology and criminology, sexual predators have been researched heavily for a long time, mainly in the context of offline predatory activity, but also with several frameworks either specifically for online grooming or agnostic to the context of the grooming.

The earliest work in Sexual Predator Identification was not necessarily aware that it would become part of the same field of study, nor was it a coordinated effort, but instead smaller clusters of researchers with a shared interest in keeping the Internet safe for children. The most common theme of the early work was the shared uncertainty of whether it was even possible to detect predators with analysis or algorithms and the lack of understanding of the actual tasks and difficulties within the field. The late 2000s and early 2010s is a period of discovery for the field, laying the groundwork and the core tenants of the field.

One of the earliest commonly referenced sources in the field is the definition of a sexual predator made by Harms (2007). Defining it as "A communication process .... in order to develop relationships that result in need fulfilment." Having introduced another researcher from informatics, Pendar, to the research on sexual predation. Pendar made the first attempt to automate the detection of predatory activity, which can be viewed as the birth of the field. The highly influential publication by Pendar (2007) built the general framework we use today, and some of the procedures are still in use. In addition, Pendar presented the most pressing issues we still face today, namely data acquisition. Pendar introduced the use of Perverted Justice, a non-profit that posts online conversations between groomers and adults posing as children. In the field's infancy, it was not known if it was even possible to discern differences between predators and victims. As mentioned in the publication, it was highly likely that the topics of conversation were shared between victims and predators since they engaged in conversation with each other. To assess the feasibility of automated solutions to detecting grooming, Pendar employed Support Vector Machines, a supervised learning algorithm, and K-nearest neighbour, an unsupervised clustering algorithm, together with N-grams and BOW to prove that predators and victims use different subsets of the English language. Since PJ only has positive labels (i.e., conversations that are confirmed to contain predators), the results cannot be directly compared against today's solutions, which has both grooming and non-grooming data in the datasets. However, using the K-NN with Trigrams Pendar achieved a 0.943 F1-score, meaning that the best solution could confidently differentiate between victims and predators. With this publication, the necessary proof of concept for the field had been made.

Using Perverted Justice, with the same data source as Pendar, Kontostathis and Leatherman (2009) sought to uncover the communicative strategies of sexual predators online to find patterns through keyword matching. The publication is the first to clearly state the difference between two types of classification, predator vs non-predator

and predator vs victim. However, the publication does not leverage the differences, only noting how difficult it is to discern predators from their victims compared to grooming in a chat. In addition, being the first multi-disciplinary paper, mixing Luring Communication Theory (LCT), a model for communication processes used to analyse predation in psychology Olson et al. (2007). LCT describes five phases of grooming in the following order. Gaining access, deceptive trust development, grooming, isolation and then approach. LCT was expanded by one of the paper's co-authors to adapt it to the context of online grooming. The technical implementation of the expanded LCT included nine categories of words that described either LCT phases or subphases of LCT. Kontostathis and Leatherman made a dictionary of 454 unique words distributed among the nine classes. The dictionary was created by manually analysing twelve chats from PJ. The dictionary coded messages from victims and predators, using the count of words in each subcategory to classify them. The classifier gained a 60% accuracy, showing that victims and predators use the words from each luring class in different amounts, which means that theoretical frameworks like LCT can be implemented programmatically to detect and analyse predators. The publication is the first example of methods from outside text mining being used to analyse online predatory behaviour. One of the most valuable contributions of the work is that since each line was coded individually and could be done in real-time, it unwittingly is the first early detection system, a subfield of SPI, that has gained some traction over the last five years. However, as with the work by Pendar, this is a proof of concept, having several weaknesses, namely how fragile pure keyword matching is for text analysis, coupled with small samples for both training and test sets.

Another set of researchers leveraging the Perverted Justice data were Gupta et al. (2012), focused on using another theoretical framework, where Kontostathis and Leatherman used LCT as their base, Gupta et al. used the theoretical framework known as "the theory of online grooming" developed by O'Connell (2003). Instead of the nine classes of luring from the expanded LCT, online grooming theory operates with six stages of grooming: friendship forming, relationship forming, risk assessment, exclusivity, sexual and conclusion. The most critical characteristic of the framework is that different adults will move through the stages at different paces and in different orders, sometimes skipping parts, which means that these are general stages that most grooming cases follow. With 75 chats manually annotated for the six stages of grooming, Gupta et al. built psycholinguistic profiles for each stage of grooming by using LIWC, a text analysis tool developed by Pennebaker et al. (2015). Initially analysing the psycholinguistic makeup of each stage, and then tracking the transitions between the stages. One of the most exciting findings is that the sexual stage of grooming is neither the most prominent nor the central stage as Gupta et al. believed, but rather the relationship forming was the most prominent. The relationship-forming accounts for 40% of all messages, while the sexual stage online accounts for 24%. The analysis also proved that the psycholinguistic categories were more than capable of discerning the different stages by showing that the different stages contained different semantic and lexical content. The publication is one of the first to introduce nature describing features through LIWC, forgoing standard

statistical and lexical features. A similar endeavour was previously attempted by Wollis (2011), using LIWC with a modified version of the five stages of grooming, reducing it to three stages; the modification of the theoretical framework coupled with problems related to keyword matching with LIWC made the work less impactful than the study by Gupta et al. In later years, several new efforts have been done with nature-describing features, like behavioural features, sentiment and emotion analysis, have been used, making this the first example of such features being used to model predators, which by definition has been characterised to have a deviant personality.

3.3.2 PAN 12

PAN is self-described as "A benchmarking activity to uncovering plagiarism, authorship and social software misuse" [1]. One of their shared tasks in 2012 was called Sexual Predator Identification (SPI), and the attempts at solving the task added a large body of work to the field. The shared task standardised the framework for SPI by providing a dataset and suggesting an evaluation metric for the tasks (Inches and Crestani, 2012). The shared task was split into two problems:

1. Detect all predators from a set of chat logs with both grooming and non-grooming conversations.

2. Identify which lines were the most predatory

There were several exciting approaches to the problems introduced. Due to the imbalance in the dataset, being less than 5% predatory conversations, many participants employed filtering methods. The goal is to remove chats that cannot be predatory, in most cases becoming a trade-off between losing a few predatory chats and removing many non-predatory chats to balance the dataset better. The two most successful approaches being prefiltering and two-stage classifiers. Prefiltering is a rule-based approach, where some metrics are used to remove chats that should be disregarded. Two stages classifiers instead tried to split the problem into two subproblems. Firstly can predatory chats be detected, and secondly, given a predatory chat, is it possible to find which participants are the predators.

Most of approaches used behavioural features or lexical features to represent the text. In this context, lexical features are purely based on the text, such as BOW or LIWC. In contrast, behavioural features look at a participant's actions or the patterns in the chat log, looking at who has sent the most messages and the percentage of words a participant sends.

Villatoro-Tello et al. (2012) had an attempt that both outperformed and was radically different from other entries. Using both a prefilter and a two-stage classifier, they were able to gain an f0.5 score above 90. The novelty of the approach is twofold. Firstly the choice of metrics for the prefilter was unique. They asserted that focusing on the most important cases, i.e. removing conversations that did not contain enough information

[1] pan.webis.de

to be classified accurately or only contained a single participant. The assumption they made was that having more than five *interventions* per user would be a good indicator of having enough information to be classified. They also removed conversations with unknown characters that they believed represent ASCII art, text formed to look like an image. The prefilter reduced the dataset by 90% while keeping above 90% of the predators, which slightly increased the relative number of predators. The other novel invention was their approach to the two-stage classifier. The approach split the problem using a Suspicious Conversation Identifier (SCI) classifier, followed by a Victim from Predator disclosure (VFP) classifier.

The SCI was trained by taking all conversations with at least one predator and labelling them as predatory, while the rest were labelled as non-grooming chat and trained their first stage algorithm on those to separate predatory conversations from non-predatory conversations. A similar approach was employed for the VFP classifier, but instead taking the conversations with at least one predator and splitting them into *interventions* as they called it, one for each participant containing only the messages from that participant. These interventions were then used to train the classifier. The finished algorithm used the two classifiers in series to first find suspicious conversations, then split the text content, and find which of the interventions was from the predator.

For the PAN-12 competition, Morris and Hirst (2012) used lexical features and behavioural features. The lexical features were BOW and TF-IDF or variations of the two, combined with unigrams and bigrams. For the preprocessing, emojis, names and numbers were normalised. Examples being that emoticons were made into four classes.

In addition to this a novel feature called the partner flip was introduced. Partner flip adds the number of times the other participant has said words that occur more than ten times in total in the chat to the BOW of their partner, which means that the feature that describes a predator also contains the more common words used by the victim.

For the non-textual aspect of the study, the behavioural features were crafted using author level features like the total number of messages sent from a user. In addition, several features tried to model their actions in the form of initiations and attentiveness. The initiations tried to capture the engagement of the users, while the attentiveness was more about keeping conversations going.

The approach for the architecture was reminiscent of Villatoro-Tello et al. trying to use two layers of classifiers, where the first tried to take predatory from non-predatory, and a second classifier to do victim from predator classification. To ensure that the classifiers never classified two predators in the same chat, predators were only the one with the highest confidence score of the two participants in a chat, the other being labelled as a victim.

The results indicate that lexical features are powerful predictors, indicating that pure lexical features with partner flip were stronger than behavioural features.

Several other researchers from the competition experimented with behavioural features and LIWC with varying degrees of success. However, this shows a great deal of belief and intuition pointing towards nature-describing features as viable solutions (Parapar et al., 2012) (Hidalgo and Díaz, 2012).

3.3.3 LCT and Chatcoder

Mcghee et al. (2011) improved upon the work of Kontostathis and Leatherman by building the second iteration of Chatcoder, named Chatcoder2. The previous iteration had used phrase-matching. Having a new goal in comparing the coding done by Chatcoder against human coder, in this context being those that annotate the data in accordance with the LCT framework. The system was changed to use rule-based matching instead, which improved the system's performance and coding interoperability. Rules are more flexible than pure phrases, giving them the ability to look for all sentences that contained two-word groups or other more advanced specifications. Grooming: "A post contains a communicative desensitisation word (penis, sex)." Personal information "A post contains an approach noun (car, hotel), a relationship noun (boyfriend, date), and does not contain a personal information noun (age, pic)." In addition, the extended LCT model was reduced to use three stages named "Exchange of personal information", "Grooming", and "Approach". The change was based on the assumption that the original nine classes to be too many for short conversations. In addition to the rule-based features for coding the conversations, a set of features were made to train machine learning (ML) algorithms. Using the count of different lexical features and POS tags, an algorithm was trained to see how well they could automate the placement to different stages. The rule-based approach and the ML algorithm were tested against human coders to see which approach overlapped the most.

Based on their accuracy results, several ML approaches were close to the 65% accuracy with the rule-based system. However, on further inspection, the rules built inside the ML algorithms to determine the category used approximately six times as many rules as the original manual rule-based approach, meaning that they overcomplicated the issues for the same accuracy.

The results indicate that the ML algorithm is less capable than the new rule-based system. However, due to inconsistent coding from humans, the results are hard to verify. Regardless of the issues they faced, it is clear that any work towards mimicking human coders for annotation of grooming data is beneficial in a field where the acquisition and annotation of data is a significant pain point.

Taking the use of LCT further was Cano et al. (2014), which tried to apply a set of six different feature types to the classification of LCT stages. With BOW, POS tags, writing complexity, sentiment analysis, LIWC and discourse patterns. Cano et al. tested all the features independently and all features combined for each stage to make a classifier make three separate classifiers. Some terms were normalised to preprocess the textual data, translating chat lingo and emoticons to standard English for predators while removing stop words and stemming the words for all users. Except for in the case of Trust development, using all features were the best solution. However, all n-grams did on their own manage a perfect recall. Feature analysis was performed by leveraging the information gain from each feature. The general results being that Discourse features and all features performed best in precision, at around 80 varying for the stages. Gaining a recall of 100 overall stages and a total precision of 70, meaning an f1 of 84.7 The most exciting finding is from the discussion where the analysis of which features performed well

shows that sentiment polarity is a weak predictor. However, more fine-grained sentiment in the form of emotions is a promising predictor.

Diverting from the attempts of using different feature types Kim et al. (2020) used sentence embeddings with LSTM, with the coded data from Chatcoder 2 and PAN-12 to train a model to first label the data into the three categories from Chatcoder 2, approach, grooming and exchange of personal information, in addition, using a second classifier to determine if the chat containing those message categories would be predatory. The thought is that since predators and victims have different ratios for the LCT categories, that can be used to categorise if the participant is a predator or victim. To train and validate the model, they split a custom made combination of PAN-12 and Chatcoder with an 80/20 ratio. The model gained a precision of 0.9063, recall of 0.9355, F1 score of 0.9148, and F0.5 score of 0.9058, which would comfortably put it as a top performer in the PAN-12 competition and be the best by a wide margin for recall. This asserts that the method has merit, however, since they used a mixed dataset, and with just a subset of PAN-12, the results are not entirely apples to apple comparison. This is the most unambiguous indication that the LCT framework can gain as high a score as the two-stage approach, showing that several approaches are competitive as state of the art for detection of predators.

3.3.4 Early dectection

Escalante et al. (2016) Tried early text classification with four datasets, comparing the use of ensembles and SVM to determine how well the algorithm performed at a given per cent of the information. With an implementation of Naive Bayes known as Early Naive Bayes, that makes prediction for every x words, to show how much of the information in a text is needed to perform the prediction, in effect performing several predictions at different points for the same entry. To preprocess the data for classification, stop words were removed, and the words were stemmed. For SPI, a 3-gram character representation was used. The results indicate that 10 - 20 per cent of the information was needed to get 80 to 90 per cent of the performance possible. While the Early naive Bayes tops out at an f1-score of 65, which would place it in the mid-field of PAN-12, the goal of the publication was not to make the best performer for classification but to assess the possibility for early detection. Early detection turned out to be a difficult task, which was assumed to be due to the imbalanced data. The main interest of this paper being that it seems to be the first paper to attempt using early detection on SPI. It is a motivation for further work, concluding that more theoretical analysis of the problem space and proposed methods are needed to further the field.

Fellow NTNU student Kulsrud (2019) attempted to classify all three levels of predatory identification, message, conversation and author. The thesis focused on the conversation level, which was the basis for early detection. Using relatively standard preprocessing in the style of Villatoro-Tello et al., and with TF-IDF, used with four classifiers, the results mostly showed that the Ridge classifier and SVM with TF-IDF generally performed best across the board.

On the Message level classification, the F0.5 score was around 32, which was self-

described as not good. However, this is a complex task, given that predators write perfectly normal messages and predatory messages. Regardless of the somewhat lacklustre results on detecting grooming in messages, the conversations and authors were detected. The results were competitive, gaining at best a 90 in f0.5 with SVM (TF-IDF) followed by a Naive Bayes (TF-IDF) in a two-stage approach. However, the most exciting experiment was the attempt at finding how many messages were needed to classify a predatory conversation, turning out to plateau with close to 95 per cent of the performance at about 20-30 messages. As pointed out in the discussion, this points to the early phases of chatting having all the textual information needed to detect grooming, which would be somewhat contradictory to some beliefs regarding stages of grooming and when different parts of grooming are performed.

Building further on this is the thesis by Vogt et al. (2021), who proposed early risk detection for SPI. Using state of the art transformer networks in three BERT dialects, a two-tier approach analyses sliding windows and continuously classifies the sequences within the sliding windows. Vogt had the same issues that Kulsrud had, finding the sweet spot between performance and early warning. The metrics used to evaluate how good something is at early prediction are combined metrics using accuracy, speed, and F-latency. F-latency uses how many messages have been sent before the warning is given and how accurate the warning system is. Both of the classification steps are simple in concept, the first only using BERT, Large, base, or mobile, which makes the features themselves, to classify if a window of size x is predatory. With all these predictions for a series of windows has been performed, a set of ten continuous windows are classified as either predatory or not, based on a threshold called scepticism. The approach reached an impressive F-latency of 0.81, indicating that the system both predicts early and accurately. For real-world implementation, a system with such a high score is viable for use in the state that it is, motivating that it is possible already today, without further data to make solutions that have a possible chance of processing large amounts of data for LEA.

3.3.5 Nature describing features

Nature describing features seek to profile the person's traits or patterns in their actions, rather than some statistical part of the textual data. The reason for making this separation is that several researchers point out in their discussions and introductions that there is an emotional, sexual and psychological difference between predators and other people, making the progression in modelling these differences an important part of the field.

One of the first uses of nature describing features was the use of sentiment and emotion-based features done by Bogdanova et al. (2012a). Using the emotions, anger, disgust, fear, joy, sadness and surprise, and their percentage of each marker as features together with sentiment words, fixated discourse, and neuroticism. In addition to this, they employed the work of Edwards et al. to find words from several categories: Approach words, relationship words, family words, communicative desensitisation, and information sharing. All these features were used on the same set of data as they used for the initial analysis,

and the results point towards higher-level features being better than pure textual features like N-grams or BOW. The High-level features gained an average accuracy of 92, while the best textual feature, char trigram, gained 72.

One of the earliest attempts at modelling the behaviour of predators comes from the second work of Bogdanova et al. (2012b), using fixated discourse through chains of semantically related sex-related terms. With the word "sex" as start points for chains and looking for semantically similar words with the two similarity metrics, the first one was devised by Leacock and Chodorow, while Resnik made the second. The similarity measures made it possible to generate chains of related words. Bogdanova et al. believe this is an adequate way to model the unwillingness of a predator to change the topic from something sex-related. The chains were evaluated on three datasets Perverted Justice, NPS Chat, and a sample of 34 logs from oocities, a cybersex page. The three datasets had significant differences in mean length of sex-related lexical chains, showing that cybersex is mostly only sexually related and has very long chains of between 12 and 18 words, while PJ had shorter chains in the range of 8 to 12 NPS had between 0 and 6. The exciting thing here is the ability to take some behavioural feature of predators and model it to a specific feature that is analysable.

Bogdanova et al. (2014) Using a selection of high-level, the main contribution of the publication was the feature analysis. These features were emotional, the word categories from Mcghee, neuroticism, fixated discourse and others (emoticons and imperative sentences). The paper shows the results of testing five individual high-level features on an SVM and combining all features with the hold-out of a single feature at a time. These two setups make it possible to discern what that feature added to the text representation. The emotional features and fixated discourse was shown to be the most discriminating features, while neuroticism and "other" were shown to be the weakest. The point is proven by the emotional features having the most impact when removed from the text representation.

The best solution is to use everything but the neurotic features. It increases the accuracy to 0.97. The best performing features are SVM (char trigram and high-level features) at 0.97 and 0.94, respectively. Also, testing single feature groups, emotion features outperformed for discerning between sexual talk and predatory behaviour. The exciting note made is that emotional features are specifically powerful at separating sexual talk from grooming.

One of the earlier papers by Morris and Hirst (2012) to use behavioural features found that at least 0.56 could be attained with pure behavioural features, meaning that it would be considered a mid-field entry in the PAN-12 competition. The performance indicates that pure behavioural features can get an outstanding result on a highly imbalanced dataset. They also mention an interesting theory, that 1% of the features capture close to 99% of the information, motivating further research into the information gain of features.

Cardei and Rebedea (2017) attempted to implement imbalanced learning together with behavioural features and textual features. The thesis had the two-stage classifier as the basis for the approach while making some crucial additions. Firstly, the parameters for prefiltering were adjusted to only use conversations with two people and more than

twenty messages. The textual preprocessing of the dataset included stop word removal and pruning words that are longer than twenty letters. A BOW-BIN was used together with behavioural features that are based on Morris and Hirst (2012), using initiation and responsiveness - a renaming of attentiveness. In addition, several features, like negation rate, ease of reading with Flesch reading ease score, and sexual/slang word ratios were employed, and the partner's values for these features were also included when generating the features for VFP identification. The two classifiers that were tested were SVMs and RFs, both wrapped in the MetaCost algorithm to handle imbalance better. The final two-chain classifier with all features used gained an f0.5 of 0.957, meaning this is the best result at the time of publication and almost to date. The most exciting part of their work is the analysis of which classifier performs best with which features. An f0.5 of 0.957 is hiding some way more interesting findings, and that is the 100 Recall, 93 Precision of RF with only behaviour features for the VFP module. The SCI module has a mediocre performance of 0.938 for all values, which means that the results are primarily coming from the module that usually is the weakest performer of the two. What is the most interesting is that the results for the VFP were obtained purely with non-lexical features, showing that modelling the nature and actions of the predators in some cases outright outperforms lexical features, which contradicts findings from some other publications. The results must be read with caution due to the metaCost balancing. Since all results VFP have the same cost matrix, there is no way to infer what comes from the features or what comes from the meta cost in comparison to a general random forest. They suggest future work with sentiment analysis and LDAs, but what is most inspiring is that they managed to improve the two-stage classifier by using behavioural features, showing that they most likely can leverage information not present in the lexical features.

A unique study was performed by Cheong et al. (2013), in a research partnership with a Danish company known as MovieStarPlant (MSP). Using a unique dataset from the children's game with the same name MSP, they tried to make a detection system to find sexual predators in a real game. With a combination of lexical, sentiment and behavioural features, with and SVM. The results were quite good results on their dataset, which in contrast to most others, were relatively balanced in terms of information from each group, 40 k vs 60k. Their ML solution outperformed all blocklisting techniques for finding predators, and they did find that BOW approaches performed very well when it was clear predatory intent. However, their behavioural features were almost as good, and they saved much computational time. The main contribution of this paper was to show that it is feasible to use real predators to detect sexual grooming and that it is a solvable problem. They tested the trained methods on the PAN-12 tests and were able to gain a 0.78 f1 score, with an accuracy of 93%, meaning that there must be significant overlap between real predators and adults pretending to be predators.

Pranoto et al. (2015) analysed the value of 20 different behavioural features to be able to find the most independent ones, using the independence from others to indicate their predictive power. Building on several theoretical frameworks like the online grooming theory, he constructed the 20 features that would capture aspects of grooming and chose the five with the strongest predictive power. Using these features with TF-IDF and LR,

he gained a 94% accuracy. Even though this is a newer work, it used PJ and Literotica to compare the sexual aspect of chatting. In this case, the features are also unique, being even more high-level than most other features that try to capture behaviour. One example is whether a predator asks for information about the child's parents as a binary feature. A feature type that is not present in other publications but found here is the attempt at modelling fantasy, which is an integral part of sexuality and grooming. There is a larger body of work related to the difference in contact-driven and fantasy-driven sexuality, also specifically for grooming.

3.3.6 Deep Learning

One of the first attempts at using Deep Learning (DL) techniques in SPI was by Ebrahimi et al. (2016). They employed single layer Convolutional Neural Nets (CNN) and several textual features to classify predatory chats. Their main contribution was to study how to implement CNNs for SPI and which textual features were most fit to use with CNNs. Testing the Neural Net (NN) with one-hot encoding, BOW and word embeddings, the best results were achieved with one-hot encoding, having an f1-score of 80%. Even though the performance does not outclass other work, they discovered that general word embeddings had subpar performance, and single layers of NN had in general better performance, maybe indicating that the textual feature extraction they did lacked sufficiently complex information to process with the current text representation. They encouraged future work on the use of LSTMs rather than CNN, which is what Liu et al. (2017) attempted.

Approach the SPI with sentence embeddings and LSTM-RNN Liu et al. achieved what would be the best results for SPI at the time. Approaching the problem by first generating language models, that in turn, generated sentence vectors, which would then be fed to a two-layer LSTM to detect suspicious conversations, and lastly to a FastText based classifier, which scores based on sentiment analysis to find which participant is the predator. In contrast to most other approaches, they chose not to filter the dataset but rather to preprocess the text to normalise it, removing special characters. What discerns this approach from the others is primarily that they have performance that rivals Villatoro-Tello et al. Slightly exceeding both in recall and precision during training, this is a more complex solution, but it does not require the manual filtration of chat logs, which makes it much more versatile.

3.4 State of the art

This section is dedicated to the state of the art solutions to detecting predators. The section will present the approaches that have the highest performance or is considered a leading publication for another reason.

The first publication that can be considered state of the art (SOTA) is the highly influential publication by Villatoro-Tello et al. (2012). However, this publication is SOTA mainly because the approach and assumptions have become the closest to a standard approach for the field. Expanding on the Villatoro-Tello et al.s approach is Fauzi and

Bours (2020) which tries to leverage ensembles and several lexical features to improve the performance within the same framework—experimenting with a collection of seven algorithms and four text representations, in addition to building ensembles of the best combinations. The approach added an element from Morris and Hirst, assuming there will only be a single predator and therefore only setting the most confident predatory pick from each conversation as a predator. The main finding aside from the SOTA performance was that their soft voting ensembles, in general, perform best of all classifiers on finding predatory conversations, gaining a perfect precision, recall of 0.954, and an f0.5 of 0.9904.

The voting ensemble they created had three classifiers combined in both a hard or soft voting combination. They proved that Villantoro-Tello s approach can perform even better, meaning that it is possible that relatively "light-weight" solutions can still be among the top performers in the field.

Coming from the same group of researchers Wani et al. (2021) has developed an entirely different approach. Instead of using filters and two-staged classifiers, a novel approach is presented. Using the most prevalent words exclusive to victims and predators, they generated a vocabulary of the most common words that separate the groups. In addition, they expanded upon EmoLex, which is a lookup-table of emotion words, to generate the new MoodBook lookup-table. Leveraging the fact that predators are emotionally different from other people, usually in the form of being emotionally unstable. With those two features, they tried to predict predatory authorships without filtering on suspicious conversations.

The approach proved promising, reporting performance close to 0.96 in all performance metrics when using a RandomForestClassifier. What sets this paper apart from most others is that they could find a pairing of features that are discriminating enough to find enough information in the text to separate groomers from non-groomers. What sets this approach apart from earlier SOTA approaches like Cardei and Rebedea and Liu et al. is that the performance is across the board, not just very high in precision, but both precision and recall. This makes the newly proposed solution much more versatile.

Lastly, there is a state of the art approach that is unique in the sense that it tries to encourage work in another part of SPI. Lykousas and Patsakis (2020) presented a new unlabelled dataset containing four years worth of live streaming data from LiveMe, which has been verified to contain grooming attempts. The main body of the work is two-fold, firstly presenting how the data was collected and the access to this data, and secondly an analysis of the data which shows that unsupervised learning on unlabeled data can detect grooming.

The detection of grooming was done by using the sexual terms in the LIWC dictionary and finding the most related terms via the FastText word embedding. Taking all sexually related words and replacing the tokens with a singular SEX_TERM token, the researcher was able to find words that appeared in contexts with the SEX_TERM. Through analysis of the words and sentences associated with the SEX_TERM, words like "open" and "show" were prevalent, in addition to emojis like the bikini emoji.

This approach is novel in SPI, and it motivates that it should be possible to find

grooming in datasets that are not labelled for it. This is a considerable advantage in a field that has been somewhat stifled by the lack of access to larger quantities of data and real data.

4 Data

This chapter explains the types of data needed to classify and analyse predatory data and the datasets found during the exploration of the field. Firstly the chapter explains what prerequisites exist for this type of data, followed by descriptions of all datasets that were considered, ordered by if they were obtained or not. Lastly, the chapter describes processing done to make a data exchange format for standardisation of the data.

In the field of SPI, getting access to datasets is one of the main challenges. Since we are dealing with children, there are several legal and ethical matters one needs to consider. Grooming and sexual abuses are both taboo and illegal activities; therefore, Law Enforcement Agencies (LEA) restrict access to all data containing actual examples of grooming. The reason being that victims are a vulnerable group. Ideally, researchers would want to have datasets containing messages from both victims and predators to get realistic results.

Today there are several datasets available to LEA and researcher, as described by Keyvanpour et al. (2016) most the conversations are between participants from the following groups:

1. Victims (children that have been groomed)
2. Psudo-victims
    a) LEA pretending to be a child
    b) Adult volunteers pretending to be a child
3. Adults chatting
4. Children chatting
5. Adult consensual sexting
6. Predators

Most datasets for grooming analysis and prediction will contain some balance of these classes. To discern different types of relationships from each other, we need examples of several types of interactions. Grooming is, in most cases, represented by messages between pseudo-victims and predators. Sexting between consenting adults represents the difference between sexual language and grooming. Lastly, general chatting between adults and children is the neutral baseline used as a comparison.

In addition to the different types of participants, we also distinguish between labelled and unlabelled data. Labelled data is data where we know what the contents of the data

represents; in our case, it could be knowing if a chat contains a predator and which of the participants is the predator. Unlabelled data would be only the chat-log, without any knowledge of the classes or participants.

4.1 Datasets

Datasets are data collections with some shared attribute, i.e., the data is from the same source. This section describes all the datasets that have been obtained or considered for use as part of the Thesis. Each dataset is described by the sources it was collected from. In addition to the data format, characteristics of the dataset, and any limitations of the dataset.

4.1.1 Pan12

The *PAN12* dataset was made available by PAN in 2012. PAN is a series of shared tasks in digital text forensics started in 2009 and held as a part of the Conference and Labs of the Evaluation Forum (CLEF) and hosted by The Web Technology & Information Systems Network (WEBIS). Pan states that they are "... a series of scientific events and shared tasks on digital text forensics and stylometry". The dataset was released as a part of the competition to further research in the field of Sexual Predator Identification (SPI). The goal of the competition was to solve two tasks.

1. *Identify the predators among all users in the different conversations*
2. *Identify the part (the lines) of the conversations which are the most distinctive of the predator behaviour*

The dataset consists of messages from several sources, including chat logs from open chat forums like Internet Relay Communication (IRCs), services to meet new people like Omegle and interactions between pseudo-victims and predators taken from the Perverted Justice Website (PJ)[1].

IRCs are large communities of people chatting on text-based servers. These servers support both private one-on-one messaging and public/private chatrooms called channels. The channels are topic-based, so the users can subscribe to topics that interest them. A user can be part of the #main channel, but not the #GameOfThrones channel. By crawling the public channels, researchers and others can collect large amounts of text data. Geeks or tech-savvy people are the predominant userbases for IRC because the IRCs are not as plug and play as more moderns solutions like Discord[2].

Omegle is a website that allows two strangers to connect and to have an anonymous online conversation. A user finds a partner by choosing a topic, and the user will be matched with another user who has chosen the same topic. When these documents were extracted, this was a text-based service akin to IRCs. What separates Omegle from

[1] http://www.perverted-justice.com/?con=full
[2] https://discord.com/

regular chatting services is the ability to disconnect from a partner without warning. Unless personal information was exchanged, neither party would ever be able to track the other one down. This feature impacts the progress of the chat, seeing as Omegle often has short conversations, where one part has failed to pique their partner's interest.

Perverted-Justice (PJ) is a foundation that had the goal of protecting children online. They trained adults to act as pseudo-victims. Besides awareness, the main activity was luring predators to commit crimes on record. The famous show *"To Catch a Predator"* was a part of PJ, where they cooperated with LEA to meet and detain a predator. Every online chat that ended in a conviction would be made publicly available on their website.

Most of the dataset contains general Internet communication taken from the IRCs. These are conversations from an extensive range of topics. The only common factor is that there are no messages that can be interpreted as grooming attempts or sexual conversations. The reasoning for this is to have a baseline with regular conversations that are not attempts at grooming. From Omegle, most of the communication was adults sexting, with the intended goal to teach algorithms the difference between sexual conversations and grooming, as stated before. Lastly, there are the conversations from PJ.

The distribution between these three types of conversations is very uneven to simulate the real world since there are very few grooming cases in the real world compared to the number of conversations on the Internet. PAN-12 chose to have 4% true positive labels, indicating actual predators. The number was chosen based on assertions made by a study of queries in peer-to-peer networks that 0.25% of queries were predatory, as found by Latapy et al. (2011) and referenced by Inches and Crestani (2012).The increase in percentage comes from the fact that 0.25% would be so low that machine learning algorithms would not make accurate predictions.

The dataset as a whole contains a mix of all these conversations, structured as an XML document in the following format.

```
<conversations>
    <conversation id="ID">
        <message line="Line">
            <author> </author>
            <time> </time>
            <text> </text>
        </message>
    </conversation>
</conversations>
```

The dataset has two primary limitations. The first one is that all the sexual conversations are between adults, not teenagers or children. Adults tend to express themselves

differently, making it easier for the algorithms to discern what is grooming and sexual conversations by "guessing" the age of the participants.

Another major limitation is that the IRCs mostly contain a specific user group, geeks. They usually talk more about computers or things related to IT or fantasy. The limited scope of the conversational material can again make it so that every message related to IT will be flagged as non-grooming.

4.1.2 Locard

Over the last few years, streaming platforms have become increasingly more popular. Streaming is where one person is broadcasting a video live stream of themselves. These streams can be video games, live discussions, or other content that people enjoy watching live.

This trend is also present among young females. They are streaming themselves gaming but also streaming other content. A particular class of this is where the streamer allows the watchers to pay money to decide what they should do. The combination of a public forum where viewers can request actions from a person, with the increasing popularity among young girls, has raised concerns about grooming.

LOCARD is a European research partnership that tries to collect digital evidence of crimes or misconduct online. One of their projects was to collect data from the Live Streaming Service (LSS) LiveMe. `liveme.com` and try to find groomers trying to solicit lewd actions from young, primarily female streamers.

The dataset comprises 39, 382, 838 chat messages exchanged by 1,428,284 users, in the context of 291,487 live broadcasts during a period of approximately two years, from July 2016 to June 2018 (Lykousas and Patsakis, 2020).

The data does not contain any labels. The researchers that created the dataset analysed it and found grooming in the dataset. In addition to this, streams often contain a lot of content and creator specific language and imagery. These expressions can range from vocabulary that is only found with a single streamer to emojis, images, or other media with a significant presence within a small community.

The dataset is formated as a .feather file, a lightweight data format meant for data frames in R and Python. The files are rows of with the following fields:

- from_user
- timestamp
- message
- ID

The one main limitation of this dataset is the fact that LiveMe has their own automated anti-grooming systems. They remove alot of the words or messages that can be seen as grooming attempts. The viewers have found ways around this by wilfully misspelling words. This is a challenge for us, since we might in some cases need to decode the misspelled words to be able to properly gain insights from the messages.

4.1.3 Twitch

Twitch.tv is a Live Streaming Service (LSS) that predominantly streams game content. The platform is also by far the largest LSS to date and is partially responsible for popularising the streaming medium. The chats from game streams can often be viewed as a conscious stream of related messages, all about the same thing. The messages are reactions to what is happening in-game or actions done by the streamer. The researchers collected the dataset to analyse and understand the differences between LSS and other types of social media. These game streams contain the same type of creator and content specific lingo that other LSS also have; however, game chats often have an even more significant amount of rich semantic content in the form of ideograms like emojis or emotes.

The data set is unlabeled and consists of the top 20 most popular games and channels between June and October 2019. Any stream that started with these games or from these channels that primarily contained English messages were collected. The collection resulted in almost 2000 streams from 666 streamers and over 60 million messages. Even though the data does not explicitly state that it contains grooming, such a large dataset should be analysed to check if it is possible to find grooming data to analyse and predict in datasets not explicitly catering to the field of Sexual Predator Identification. Being another LSS, it also should give some insights if compared against the Locard dataset. The dataset contains two files for each stream. The first file contains the meta-data collected from the stream. While the second file contains the chat log that was harvested.

The meta-data was structured as a JSON file with the following fields.

```
{
"user_view_count": 84330058,
"user_broadcaster_type": "partner",
"stream_game_id": "2748",
"stream_type": "live",
"stream_viewer_count": 3265,
"stream_start_date": "2019-09-23T22:58:05Z",
"stream_language": "en"
}
```

while the chat was formated as CSV files with the following fields:

- Time
- User
- Message

What sets this data set apart from the two others is how homogeneous it is. It only contains messages about games and is also in opposition to LOCARD, a dataset where the streamer decides the actions and controls the topics discussed.

Name	Description
Exchange of personal information	Messages about topics related to relationships, living situations, upbringing, and personalia
Grooming	Discuss sexuality or implies wants or intentions to have or partake in sexual activities with the victim.
Approach	Isolating victims from their network, getting contact information or planning meetings.
Lines containing none of the classes	General chatting

Table 4.1: Description of main classes within Luring Communication Theory

4.1.4 ChatCoder 2

The Chatcoder project is a generational project that attempts to use Luring Communication Theory (LCT) as the basis for building rule-based models to detect grooming attempts and the patterns in grooming.

Chatcoder, the original project, collected 228 chat logs from the Perverted Justice (PJ) website to develop an LCT program. Using 12 chats, the researcher generated a dictionary of words categorized into eight classes that explained different actions associated with different aspects of LCT. The second project labelled all the lines of 50 chat logs with four different classes, as described in Table 4.1.

As with PAN-12, this dataset is built as XML documents. There are three separate documents, one containing all the chats collected by Chatcoder, including the unlabelled ones, and two containing the labelled chats. The labelled chats have been split into only victim lines and all coded chats.

```
<chatcoderadmin>
    <ChatLog>
        <name> </name>
        <CodingVersionID> </CodingVersionID>
        <lineNum> </lineNum>
        <category> </category>
        <userID> </userID>
        <dateTime> </dateTime>
        <body> </body>
    </ChatLog>
</chatcoderadmin>
```

4.1.5 Acquisition of datasets

Acquisition of high-quality datasets has been a recurring issue in the field. So for ease of access acquisition methods, the datasets have been listed, with citations to the paper where they were initially presented.

Pan-12 The PAN-12 dataset has historically been controlled by the Webis foundation, which required contact by email with a request to access the data or implement a program to access the data online through an available endpoint. However, recently they have transferred the data to Zenodo[3], a dataset hosting website used by many researchers. The dataset is restricted, so a request must be made to access the data. The overview explaining the contents of the dataset has been published as a paper by Inches and Crestani (2012)

Locard Locard hosts their dataset on Zenodo, also having it restricted, so it requires a manual review before being shared. It can easily be accessed by sending a request to the maintainer of the dataset via the Zendodo[4]. The analysis and explanations for the dataset can be found in the initial paper Lykousas and Patsakis (2020)

Twitch The Twitch dataset, also called Twitch-Chat, is accessible through a page known as OSF, where they host the code to reproduce their results, together with the paper and the dataset. [5]

ChatCoder2 The author has to be contacted directly via email to access the Chat-Coder2 dataset. Currently, the author works at Us Naval Academy, where the contact information can be found. All relevant information about the dataset can be found in the paper Mcghee et al. (2011)

4.2 Unobtained datasets

This section presents the datasets that were not accessible but were of interest. Firstly explaining some general information about the dataset and why it is desirable, and then explaining what prevented the acquisition of the dataset. The list is not an exhaustive list of all datasets used or considered in the field, only those deemed most interesting during the literary search for this thesis.

4.2.1 MovieStarPlanet

The first and most desired dataset would be MovieStarPlanet. MSP is a social children's game developed in Denmark, where the children create their custom avatars. Players can play minigames or chat with people, which will, in most cases, be strangers. Groomers have plagued MSP for many years, and in 2016 a researcher Cheong et al. (2013) got a partnership with the company owning the game. The research partnership resulted in a dataset containing all messages sent on a British game server over the span of 15 minutes. In total, they collected 60 000 lines of text. In addition, the moderators of the game had already caught several groomers and flagged their usernames. They gave all

[3] https://zenodo.org/record/3713280#.YgZrfO6ZNhE
[4] https://zenodo.org/record/3560365#.YgZsXe6ZNhE
[5] https://osf.io/39ev7/

messages sent by these 60 offenders, and this totalled 40 000 messages. MSP is one of only two datasets encountered that contain actual victims, making it very desirable. Due to the research partnership, the data cannot be used outside that context. To confirm that the data was restricted to the project, the researcher in charge of the project was contacted, and they confirmed that the data would not be possible to acquire.

4.2.2 Surete du Quebec

For his master thesis in Online Predator Identification, Mohammadreza Ebrahimi got a small sample of real conversations from the Surete du Quebec, the LEA in Quebec, Canada. Ebrahimi et al. (2016) This dataset mainly contained true positive entries since it was taken from actual police records. Most of the conversations were in french, meaning that it would not be feasible to integrate with an English dataset. There is not stated anything about the availability of the dataset, so it is safe to assume that it is not meant to be distributed.

4.2.3 Perverted Justice

The Perverted Justice website has 630 publicly available predatory conversations. These are a superset of the true positive entries from the PAN-12 dataset and ChatCoder 2. However, this is not a proper dataset but rather a collection of available data that has not been collected or processed to a standard format. The conversations on the website are structured but not organized. Every conversation consists of entries or messages, and each entry has the following: text, timestamp, username, (optionally) comment. However, the ordering of these parts of the entry is not consistent across conversations, meaning that it requires semi-manual work to extract the data in a structured fashion. In addition, they post comments from the pseudo-victim in charge of the case appended to the text bodies of some entries. Usually, the comments will express disgust regarding the actions of a predator or elation over getting confessions. There is a feature on the page to disable the comments. However, the feature does not work consistently with web scraping frameworks. Sadly the comments are neither uniform nor have an exclusive pattern, so they cannot be easily removed with tools like Regex or other pattern matching without at least some loss of conversational information. The data was not collected primarily because of the amount of work required to access one dataset when about half is already available through other datasets. If one needs to boost the number of unique predatory chats, it should be possible to access them with web scrapping and some manual work. Pan12 uses about 330 positive examples between the test and training set, allowing a doubling of predators if one extracts all the predators from PJ.

4.3 Data preprocessing

Data preprocessing is the task of filtering and refining the data to be better suited for the project. This section describes the preprocessing of all datasets done prior to the analysis and prediction. Having four different datasets with different formats is a way

to introduce errors and inconsistencies to the research. The data was normalized by transforming all the conversations to a JSON format. This format enabled indexing of all conversations based on UUID of the conversations.

```
{    "type" : "predatory",
     "participents" :["id","id2"],
     "Source" : "ChatCoder 2",
     "uuid" : "stsa-sadsa-22-13-asd-sd",
     "chatMessages":[
          {
          'sender':"id",
          "line":2,
          "time":"2016-01-29",
          "text": "Hi i am a predator:",
          "?category": "Initation-pred"
          }
     ],
     "amountOfTokens": 201,
     "chatLength" : 42
     }
```

To reduce the size of the data object, all message data that was not text, sender, timestamp, or category of content for the text were discarded.

In the end, each conversation had the following information, the participants, the number of messages sent, the number of words in the text, the source, and if the content is known to be predatory. During the processing, some messages were removed. Discarding messages with a malformed text body or with invalid time information. The filtering resulted in removing close to 1 000 messages out of several million. With a standard data format, the rest of the Thesis could be written with standardised data functions, enabling the analysis and prediction to be used with all the datasets.

The datasets were individually filtered, each by some metric inherit to the characteristics of the dataset. Twitch and LiveMe by language and converastion length, since there is no guarantee for language in unlabelled dataset from open chatrooms, while Chatcoder and PAN-12 stayed untouched.

5 Architecture

This chapter will describe the final architectures of the subsystems used to predict predators. The architectures are high-level descriptions of the systems. Since each system was subject to experimentation, this chapter will only state the final configurations, while the next chapter outlines the experimental setup and how the final configurations were reached. The chapter describes the whole disconnected system and then gives a more in-depth description of each subsystem.

5.1 High-level description

This section explains the flow from initially getting the data to the finished prediction, starting with a high-level overview of submodules the system is made of and what they are meant to accomplish. Figure 5.1 shows the architecture for the whole system. There will be a dedicated section following this section to explain each submodule.

As explained in Chapter 4 there are several datasets with different formatting. Before any prediction can be made, the initial data processing is performed to standardise the format. Following this, the data is sent to the two separate systems for predicting predators.

The pipeline is split into two subsystems to test the two state of the art architectures. The two-stage classifier is a prediction pipeline fashioned after the system built by Villatoro-Tello et al. (2012) while the Social Behavioural Biometrics (SBB) classifier tries to use the emotions of the predators to predict predatory behaviour in one-stage, based on the architecture from Wani et al. (2021).

5.2 Two-Stage Classifiers

As the name implies, it uses two classifiers. The first classifier, Suspicious Conversation Identification (SCI), finds all chats that contain a predator. The first classifier acts as a filter for the second classifier. The second classifier, Victim from Predator disclosure (VFP), then uses the suspicious conversations, splitting the conversations into documents for each participant and classifying each participant from the chats as either victim or predator.

5.2.1 Preprocessing

The data from the PAN-12 competition contains a significant amount of non-grooming data. Large amounts of the chats contain small amounts of data, primarily due to the

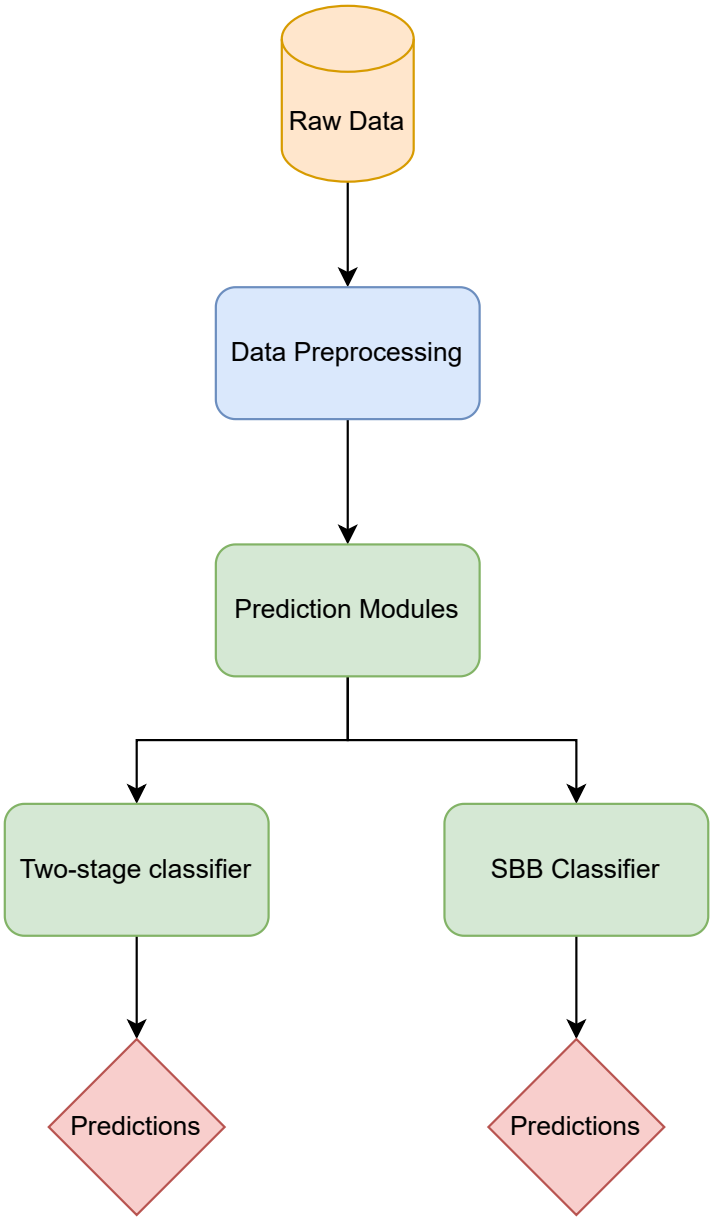


Figure 5.1: Architectural overview of system

nature of the Omegle dataset, as described in the Section 4.1.1 about the PAN-12 dataset, containing many short message histories due to early disconnects. Since the small amount of data makes the classifiers unable to function correctly, this is seen as outliers or noise. To account for the noise and outliers in the data, every chat that either contained fewer than 20 messages or had only a single participant was removed.

This prefilter removed about 80% of the conversations in the dataset as seen in Table

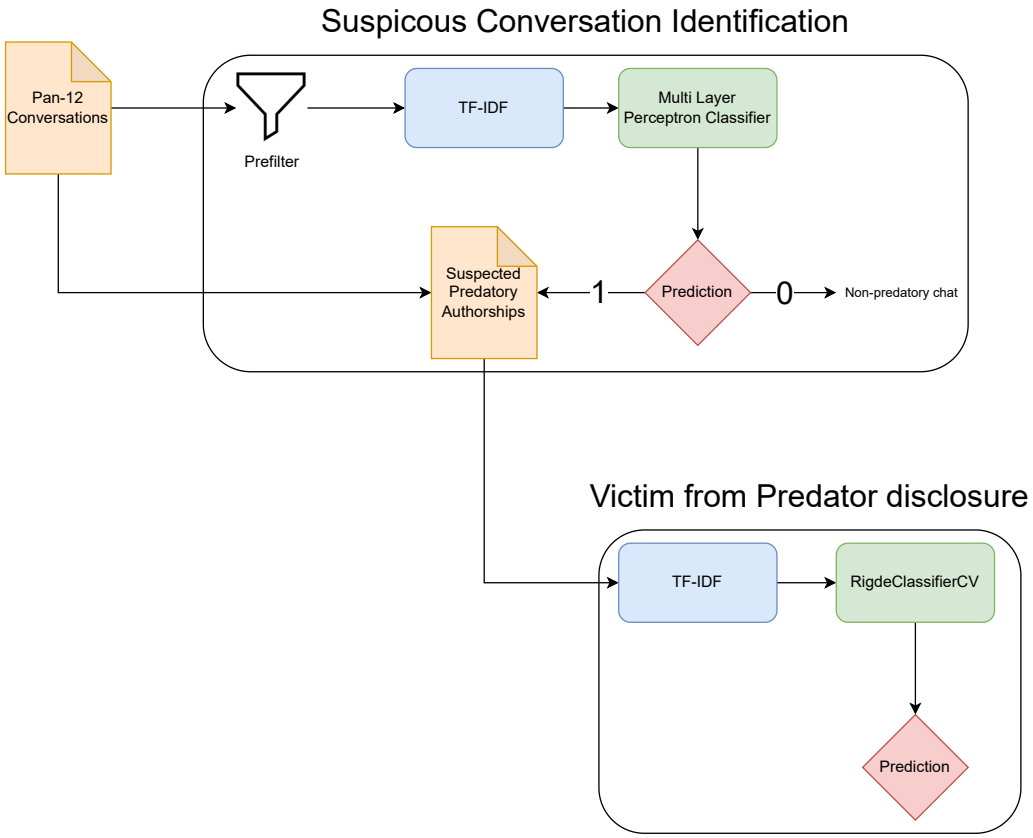


Figure 5.2: Classifier based on the architecture proposed by Villatoro-Tello et al. (2012)

Table 5.1: PAN-12 Filtering results

Filtering	Train	Test
Unfiltered	66927	155128
Prefiltered	12276	28559

5.1. For most tasks in NLP, we want to clean the textual data in some way. The goal is to normalise the data by removing information that does not help solve the task. However, in Sexual Predator Identification, this is not always desired. Primarily this is because textual cleaning, in many cases, is a trade-off between quality and quantity, where one often discard some information to remove noise. If the data is uniformly discriminator or there is uncertainty about which text features are discriminatory, we risk removing or altering text containing essential information for this specific task.

5.2.2 Suspicious Conversation Identification

Figure 5.2 has a submodule called Suspicious Conversation Identification (SCI), which will be explored further in this subsection. Prediction on textual information requires the data to be formatted correctly. In the case of conversation detection, a single document represents a chat log, and all the messages from one chat are concatenated into a document to accomplish this. Each of these documents is labelled based on whether a predator is present in the chat. This step is required since we only know which chatters are predators, not which predatory conversations.

The concatenated documents are then processed by feature extraction for SCI, Term Frequency–Inverse Document Frequency (TD-IDF). The best performing classifier for the SCI module was the Multi-Layer Perceptron Classifier (MLPClassifier), which was combined with TF-IDF to get the best performance. The trained classifier gives predicted labels to each document, allowing us to predict suspicious conversations.

5.2.3 Interlude

In between the two classifiers, there is a data wrangling step depicted as the "Suspicious Conversation Authorships" in Figure 5.2. Given that the document representation for the PAN-12 dataset is made to be a document per conversation for the SCI, it will have to be transformed to be a document per author for the Victim from Predator disclosure module (VFP).

Each conversation is then connected by their UUID, a unique identifier for each conversation, and predicted status. For each conversation suspected to be predatory, the conversation is fetched from the original data source. Then each author has all messages that are part of a suspicious conversation concatenated to a "suspected predatory document".

5.2.4 Victim from predator

Having the documents at a per author level, we can now detect if an author is predatory. The Victim from Predator discolure module (VFP) is shown in Figure 5.2, and is quite similar to the Suspicious Conversation Identification module (SCI), only using a different classifier. It is important to note that there is no preprocessing, as the prefilter and the filtering done by the SCI module is all the processing needed. For this module, the TF-IDF was still the best option. However, the RidgeClassifierCV turned out to be a better predictor than MLPclassifier for Victim from Predator disclosure.

5.2.5 Evaluation

An additional step is to evaluate the results when getting a prediction for each participant in a suspected predatory conversation. Since the final classifier from the Victim from Predator disclosure, only classified authors from predatory chats, the rest of the authors do not have a predicted value, so evaluation metrics cannot be calculated. Since every

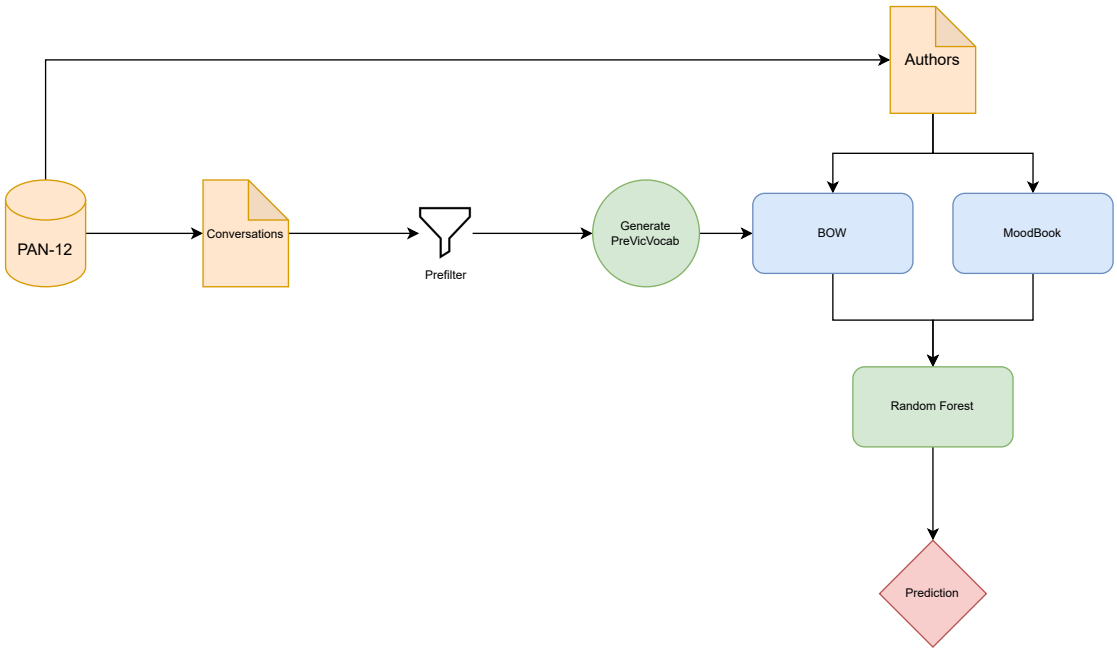


Figure 5.3: Classifier based on the architecture proposed by Wani et al. (2021)

chat that was not seen as predatory by the SCI module is stated not to be a predator, we can set all participants from these chats to non-predators.

5.3 Social Behavioural Biometrics classifiers

Social Behavioural Biometrics (SBB) is a term coined by which analyses the behaviour and interactions of people online to classify them. The goal is to make a behavioural or emotional profile for each message or chat. Differing quite considerably from the Two-stage classifier, this type of classification uses a single step, generating the documents on a per author basis from the start, instead of filtering with suspicious conversations as seen in Figure 5.3

5.3.1 Features

As a part of the feature set for the SSB classifier, a Predator Victim Vocabulary is used. Predator Victim Vocabulary (PreVicVocab) is a vocabulary defined by the N most used words exclusive by each group. The vocab is generated by first finding the N topmost used by each group, and after this, the exclusive words for one group are combined into a vocabulary.

This vocabulary is generated by first prefiltering the PAN-12 dataset, inspired by the method from Villatoro-Tello et al. (2012) but with different parameters. The filter removes conversations that are not between two participants or has less than six messages.

With the prefiltered dataset, we find the top 10 000 words from each group that have more than 24 occurrences in their respective group. Bag-of-words algorithms can then use the vocabulary as a substitute for generating the vocab by reading all the documents.

Since this is a single-stage classifier, as seen in the Figure 5.3 we split the documents on a per author basis for the classification, the same as was done for the Victim from Predator disclosure classification.

Social Behavioural Biometrics classifiers are distinct because what distinguishes the method is what features they use. They use the Moodbook together with the specialised BOW. The thought is that since predators are emotionally and psychologically different from other people, and to a greater extent, different from victims, these attributes should be used to separate the two. Moodbook is, as defined in Mudasir Ahmad Wani and Hussain (2018), a set of categories and words matching those categories, in other words, a lookup table for emotions. In addition, to the features from Moodbook, there are two final features, counting how many of the positive and negative moodbook features are present in a text.

5.3.2 Classifier

For the classifier in the SBB approach, a Labelspreading algorithm was seen to perform the best. The algorithm directly classifies all authorships in the PAN-12 dataset.

6 Sexual Predator Identification

This chapter describes the process of finding parameters for the architecture that perform well for detecting predators and the results of the approaches. The chapter first describes the shared methodology used to experiment and build the architecture shown in Chapter 5 and then the experimentation to get the final configuration for the two-stage classifier and the Social Behavioural Biometrics classifier (SBB). Following this, the results from the validation and hyperparameter tuning of the algorithms are presented. Lastly, the test results are presented, leaving the analysis and interpretation of the contents of the results to the following chapter.

6.1 Experimental Plan

In the field of Sexual Predator Identification (SPI), there is a sentiment echoed by many researchers, while no one to date has been able to provide any empirical evidence of this situation. That is the notion that we do not know the overall best feature, the best model, or the best anything for SPI. Different researchers have different approaches to solving the problem, and few studies have similar enough setups to be comparable to each other. Therefore we can not genuinely compare single aspects of the studies directly with each other without making some assumptions. The dissimilarities make it hard to conclude which specific parts of the approaches are the most important to the approach's success.

That is not to say that there is no comparison between solutions or knowledge of which features perform well for a single approach to the problem. Nevertheless, there is not a de facto best feature or model for the field as a whole.

Some researchers will find that Binary BOW (Bin-BOW) is the best approach for textual features, while others will select Term Frequency-Inverse Document Frequency (TF-IDF) for a surprisingly similar approach and find that it turned out to be better than Bin-BOW.

This somewhat chaotic element in the field requires us to make fewer assumptions when feasible without increasing the problem complexity too much. In short, this describes a problem space that requires a methodical approach to experimentation.

To this end, the experimental plan for both approaches to the SPI architecture use quantitative data to make decisions for the study, rather than conventional wisdom or heavy reliance on the specific insights from others.

The experimentation can be subdivided into the optimisation of three parts: choice of feature extraction, the classification model's choice, and the classification model's tuning.

The methodology for each of these parts is the same. The method starts by testing a base architecture chosen based on authoritative publications in the field. The base is tested with many models and features, using the results to determine the best algorithms for further investigation. Each of the best algorithms is assembled into ensembles that are hyperparameter tuned to optimise the performance. The problem is split into sections, solving them incrementally to avoid brute-forcing the whole solution and increasing the problem space exponentially.

6.2 Shared SPI experimental setup

Since the architecture for the Social Behavioural Biometrics classifier and the two-stage classifiers can be implemented with many different classifiers, this section explains the shared architectural design. They share the selection process of the classifiers, hyperparameters and selection of classifiers for voting ensembles.

All the algorithms were trained and tested on the same data to determine which had the best performance for the given task. Determining "best performance" is somewhat unquantifiable by a single unit of measurement, so four measures, Precision, Recall, f0.5 and f2.0, were employed instead.

Since the PAN-12 competition used f0.5 as the primary scoring metric, this Thesis has used it as the primary evaluation metric. In effect, all steps have been assessed on and tuned to increase the f0.5, and all tables are sorted on f0.5. However, due to there being some researchers voicing concerns regarding if f0.5 is the best choice for a primary metric in the field of SPI, f2.0 has also been reported.

The three most feasible classifiers for each module were chosen for the soft voting ensembles. The evaluation was done on a combination of performance, computational cost, overlapping performance with other algorithms, uniqueness and if the classifiers supported confidence.

Uniqueness is a constraint set due to several of the classifiers being presented as different classifiers while, in reality, only being variations of a single algorithm that can be reproduced with the correct parameters. An example of this constraint is that the Stochastic Gradient Decent (SGD) classifier is an SVC with an SGD learning approach, so the underlying algorithm is still an SVC.

Due to using soft voting ensembles rather than hard voting ensembles, the classifiers had to give a confidence value for their predictions since this is the value used for soft voting. The most common parameters on each algorithm are chosen to be tuned. The strategy for finding these is using online resources for each algorithm and then expanding the commonly used search space somewhat to give the bayesian search strategy more ability to explore the interactions between parameters freely.

Some restrictions apply, primarily due to some of the parameters not supporting other parameters, such that either the search spaces had to be restricted for parameters, or other parameters had to be dropped entirely to avoid failing too many training sessions.

Optuna, the hyperparameter tuning framework, was set to have 100 training sessions for each classifier. The number of sessions was chosen to give Optuna enough training

sessions to optimise but set a bit low to avoid computational costs since many algorithms were tuned.

6.3 Two-stage classifiers

This section contains all the information about the experimentation and development of the two-stage classifier that resulted in the final configuration as described in Chapter 5

To find the best combinations of features and models, all features were tested against all models to look for patterns in the performance.

The features used for the testing are bag-of-words (BOW), Binary bag-of-words (Bin-BOW), Term-Frequency (TF), Term Frequency-Inverse Document Frequency (TF-IDF), Linguistic Inquiry and Word Count (LIWC) and Combined(TF-IDF+LIWC). The reason for using TF-IDF and LIWC for the Combined feature is because, as we can see in Table 6.4a out of all the top five features evaluated by f0.5, TF-IDF had all top entries. When initially deciding which feature should be used for combined, only the Suspicious Conversations Identification (SCI) module had been developed, so the feature set was developed based on the indications from this module.

The six features were combined with the twenty-nine algorithms seen in Table 7.2 to find the optimal combination, and the 174 permutations were evaluated on all four scoring metrics.

6.3.1 Suspicious Conversations Identification

For the Suspicious Conversations Identification (SCI) module, the best combinations can be seen in Table 6.4a. Three of these classifiers were chosen as part of a soft voting ensemble. Ideally, these would be the top three f0.5 algorithms that can report confidence and that are not built on the same underlying model. The choice for models for SCI should be the Multi-Layer Perceptron (MLP), CalibratedClassifierCV (CCCV) and Linear SVC classifiers. Since these three all ranked highest on the primary metric, and all supported confidence measures. However, in the case of soft voting ensembles, the CCCV uses Linear SVC as a base_estimator for calculating confidence, meaning that the algorithms are based on the same algorithm. For this reason, both the Linear SVC and SGDClassifier can become the same as the CCCV, so to air on the side of caution, both were excluded, the Passive-Aggressive Classifier (PAC) classifier cannot predict probabilities, meaning that in the end, it uses a SVC. For this reason SVC, MLP and CCCV were the final selection used for the soft voting ensembles, as can be seen in Table 6.1 under the name SCI.

6.3.2 Victim from Predator disclosure

Building on the same principles, the combinations for the Victim from Predator disclosure (VFP) module can be seen in Table 6.4b.

The three best f0.5 algorithms were Logistic Regression, RidgeClassifier, and Multi-Layer Perceptron. The RidgeClassifier did not support confidence values needed for soft

Stage	Classifier 1	Classifier 2	Classifier 3
SCI	CalibratedClassifierCV	MLPClassifier	SVC
VFP	Stochastic Gradient Decent Classifier	MLPClassifier	Logistic Regression
SBB	K-nearest Neighbour	Random Forest	XGBoost

Table 6.1: Classifiers used for the soft voting ensembles

voting ensembles and was excluded. Since both the Logistic Regression was already a part of the ensemble and RidgeClassifiers were off-limits, the following algorithm of choice was the Stochastic Gradient Descent (SGD) classifier with Term Frequency, which became the last algorithm for the Victim from Predator disclosure ensemble setup.

6.3.3 Other test performed

Several other approaches, models, or features were explored in the early stages of experimentation. These smaller-scale experiments aimed to gain some insights into the field. However, all of them had shortcomings that made it undesirable to continue exploring them to the point of fully implementing a solution with them.

Some early testing was performed with behavioural features, but the features were perceived as fragile since both the format and metadata vary between sources and dataset. In addition, they are costly to design and implement, making them seem even more fragile than features like Moodbook, LIWC and textual features that are easily implemented and only depend upon the textual data.

Following this, some experimentation was done with transfer learning models. Sadly, the results were inconsistent, bound mainly to the random seed used on initialisation. In short, it meant that the median or average performance of RoBERTa, BERT, and distilBERT was mediocre compared to traditional machine learning when considering the inconsistency. The best single performance ever gained was from a transfer learning model. However, the average or median performance would be closer to the top 10%.

Afterwards, it was attempted to tune the hyperparameters of the vectorisers that generated the textual features. Using Term Frequency-Inverse Document Frequency (TF-IDF) and Bag-of-words, neither had any indications that there were improvements to be made. Testing with n-grams, restricted vocabulary sizes or minimum or maximum document frequencies.

The test was performed as a hyperparameter tuning of both algorithms and vectoriser and evaluated manually based on the top mean scoring values, where TF-IDF consistently beat out all other vectorisers and had no benefit from any tuning.

Lastly, tests with advanced neural nets in Keras were performed. The experiments tried to leverage the ability to find more complex patterns from Long short-term memory (LSTM) and Convolutional neural network (CNN), in addition to basic neural nets, all with TF-IDF and word embeddings. However, none of the initially tested solutions gave results that warranted any further investigation into the feasibility of this approach. The advanced neural networks had indications of slightly better performance in some

specific cases, but like the transfer learning models, it seemed to be outliers rather than consistently better performance. However, they had a much higher computational cost, making them a less feasible solution as a whole.

There is a caveat with early small scale experiments that the indications might come from inexperience in the field or just that the tuning of the approaches was not correct. These experiments are reported to give a complete overview of approaches that were attempted.

6.4 Social Behavioural Biometrics classifier

This section describes the steps to develop and experiment on the Social Behavioural Biometrics (SBB) classifier. Since the classifier is initially built to use a specific set of features, experimentation has primarily focused on choosing optimal algorithms for the problem, generating ensembles to boost predictive power, and hyperparameter tune the algorithms to optimise the solutions.

6.4.1 Generation of features

A core part of the SBB approach is the two features employed, Moodbook for leveraging the emotional differences between predators and other people, and Predatory and Victims Words (PreVicVocab) to use textual features that are strongly discriminatory each group.

The PreVicVocab was generated per the descriptions given in the publication by Wani et al. (2021). The process starts by finding prefiltering the conversations from the dataset, inspired by Villatoro-Tello et al. (2012). However, as opposed to the original publication, the parameters are only conversations with two participants, and more than six messages are considered. The data is normalised by removing stopwords and punctuation from all these chats.

From all the predatory conversations in the prefiltered dataset, two separate lookup tables contain all the words and frequencies from messages sent by predators or victims are generated. The lookup tables were set to have some constraints. Firstly, the tables only keep the 10000 most prevalent words and remove any words that occurred less than 25 times for each group. With the lookup tables completed, the exclusive words for each group were concatenated to a single vocabulary.

The procedure did not generate the same vocabulary as the original publication, getting 199 Victim words and 167 predatory words, whereas the original publication had 304 and 299 words for the groups. In effect, this meant that somewhere the generation had differed. The discrepancy prompted further investigation, where it became clear that for most words, the counts made by this approach, seen in Table 6.2, and the original publication, as seen in Table 6.3 were very close and, in some cases, even the same for both groups.

Curiously the values found in Table 6.3 give identical counts for two words, seemingly indicating a typo, and this might point towards some minor errors in the paper accounting

	Horny	Friends	Swear
Victim	2	87	25
Predator	78	124	10

Table 6.2: Attempted recreation of Predator-Victim vocabulary

	Horny	Friends	Swear
Vicitm	**2**	**2**	25
Predator	**82**	**82**	10

Table 6.3: Original data published about the Predator-Victim vocabulary

for the inability to reproduce the PreVicVocab. The discrepancy was further investigated in Section 7.1.5 of the evaluation.

After getting in touch with the authors of the original paper that proposed the approach, they agreed to provide access to the dataset they used. However, in their words, a dataset is not the raw textual data but rather the data transformed to features and labels. The dataset they provided was ready to be processed by a classification algorithm.

6.4.2 Model selection

The algorithms were tested on the dataset provided by the researchers, and this yielded the validation results seen in Table 6.4c.

The top three algorithms based on the f0.5 score would be LabelSpreading, Extra Trees and Random Forest. The LabelSpreading algorithm was discarded due to being computationally too expensive to compute. Initially, it failed to fit due to excessive RAM usage during the parallelized training session of cross-validation. However, the situation did not improve when fewer threads ran the process. The result was that the training sessions took several hours. For this reason, the K-NearsNeighbours was favoured above the LabelSpreading algorithm. The Extra Trees algorithm was also discarded favouring the XGBoost Classifier due to the similarities between the ExtraTrees and RandomForest classifiers. The two algorithms are very similar but with differing ideologies. ExtraTrees favour speed by not finding the optimal solutions, while RandomForest has a more optimal strategy. In short, it makes them interchangeable for this specific experiment.

6.5 Validation results

This section describes the results from the validation scores of the three modules used for Sexual Predator Identification (SPI). These results are integral to deciding which algorithms perform the best and should be part of the ensembles and which algorithms should be hyperparameter tuned. The best performance will be shown for precision, recall, f-scores and accuracy for a module.

However, since these are validation results, there are some things to keep in mind. Firstly, the validation results are based on the ten-fold cross-validation done as a part of

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
MLPClassifier	TF-IDF	**0.978**	0.973	0.972	**0.979**	**0.997**
CalibratedClassifierCV	TF-IDF	0.975	0.975	0.975	0.976	0.997
LinearSVC	TF-IDF	0.975	0.972	0.970	0.977	0.997
PassiveAggressiveClassifier	TF-IDF	0.975	**0.976**	**0.977**	0.974	0.997
SGDClassifier	TF-IDF	0.974	0.971	0.970	0.975	0.996
MLPClassifier	TF	0.972	0.963	0.960	0.975	0.996
SVC	TF-IDF	0.971	0.950	0.943	**0.979**	0.995
CalibratedClassifierCV	TF	0.971	0.966	0.964	0.973	0.996
SVC	TF	0.970	0.958	0.954	0.974	0.995
MLPClassifier	Bin-BOW	0.970	0.958	0.954	0.974	0.995

(a) Validation results for training of SCI Module

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
MLPClassifier	Bin-BOW	**0.957**	0.956	0.956	0.958	0.955
LogisticRegression	Bin-BOW	0.956	0.956	0.956	0.956	0.955
RidgeClassifierCV	TF-IDF	0.955	0.950	0.948	0.957	0.952
LogisticRegression	BOW	0.950	0.903	0.889	**0.969**	0.929
RidgeClassifierCV	TF	0.950	**0.965**	**0.971**	0.945	0.956
RidgeClassifierCV	Bin-BOW	0.947	0.942	0.941	0.948	0.944
MLPClassifier	BOW	0.946	0.942	0.941	0.949	0.944
SGDClassifier	TF	0.944	0.919	0.912	0.953	0.933
PassiveAggressiveClassifier	TF	0.942	0.958	0.964	0.937	0.948
LinearSVC	TF-IDF	0.942	0.941	0.941	0.944	0.941

(b) Validation results for training of VFP Module

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
LabelSpreading	SBB	**0.971**	**0.894**	**0.871**	**0.999**	**0.970**
ExtraTreesClassifier	SBB	0.970	**0.894**	**0.871**	0.999	0.969
RandomForestClassifier	SBB	0.969	**0.894**	**0.871**	0.996	0.969
KNeighborsClassifier	SBB	0.967	0.893	**0.871**	0.994	0.968
XGBClassifier	SBB	0.964	0.893	**0.871**	0.990	0.968
MLPClassifier	SBB	0.963	0.893	**0.871**	0.989	0.968
BaggingClassifier	SBB	0.960	0.892	**0.871**	0.985	0.967
ExtraTreeClassifier	SBB	0.960	0.892	**0.871**	0.985	0.967
DecisionTreeClassifier	SBB	0.956	0.891	**0.871**	0.980	0.966
LogisticRegression	SBB	0.954	0.888	0.868	0.978	0.964

(c) Validation results for training of SBB

Table 6.4: Validation results

the training of each algorithm. Meaning one-tenth of each training set has been held-out to be used as a validation set; this has been repeated ten for all ten folds, giving an averaged score for validation across all folds.

Another note that must be made is that both the SCI and VFP modules have had the six different feature combinations versus one feature set for the SBB module, meaning that they have six times as many results in the non-truncated table. The difference in results might make it such that only showing the top ten algorithms might give the impression that the VFP and SCI generally were more even in performance when it comes to the f0.5 score. However, this is more a quirk due to the differences in results per table, rather than a point of interest that should be investigated further.

6.5.1 Suspicious Conversations Identification

For the Suspicious Conversation Identification, it is clear that Term Frequency (TF) based feature extractions are favoured for this problem and that pure textual features are more suited than those incorporating LIWC. From Table 6.4a we can see that nine out of ten solutions use TF-based features, while all ten use textual features.

Another pattern is small margins between solutions, with the f0.5, accuracy, and precision having an interval of below 0.01. The margins are even more apparent with the accuracy, where the score only has an interval of 0.002, indicating that it might be a not very sensitive metric for the SCI module.

Evaluated on the primary metric, f0.5, Multi-layer Preceptron with Term Frequency-Inverse Document Frequency (TF-IDF) has the best performance, with a score of 0.978. Since F0.5 is a derivative of recall and precision with a skew towards precision, it is not surprising that this solution also has the shared highest precision with SVC (TF). A similar pattern can be seen for the Passive Aggressive Classifier (TF-IDF), which has both the best recall and f2.0. With values of 0.977 and 0.976. These are linked in the same way that precision and f0.5, making it natural to have these links between the scores.

6.5.2 Victim from Predator disclosure

In solutions with several stages of classifiers, errors and bad performance will usually propagate from earlier to later. However, a subtly in this is that such problems are only applicable for test results and not local validation results. This note has to be made since it makes it such that the validation scores seen here will not be comparable to the test scores presented in the following two sections.

In the same manner that TF-based features were favoured for SCI, we can see in Table 6.4b that the three algorithms, Multi-layer Preceptron (MLP), Logistic Regression (LR) and RidgeClassifiers, generally perform well. In addition to this, we can see a relatively uniform spread among the textual features in the top ten. These two facts could indicate that algorithms are more vital than features for this problem.

The strongest predictor for f0.5 is MLP (Bin-BOW), with a score of 0.957 in f0.5. F0.5 is the only metric where the MLP has the highest score.

The LR (BOW) has the highest precision with a 0.969, having a score of 0.01 over the preceding score. The gap is a considerable margin taking the small range of values for precision.

The RidgeclassifierCV (TF) has the best performance for the last three metrics, f2.0, accuracy and recall. Scoring, 0.965 in f2.0, 0.971 for recall and 0.956 for accuracy. The solution also has a considerably higher recall than the runner-up, with a gap of 0.015.

6.5.3 Social Behavioural Biometrics

The SBB results are unique in one fashion. Though the results are sorted on f0.5, every column in Table 6.4c has ended up being perfectly sorted in descending order. What this means in effect is that the algorithms are sorted in order of superiority, where each place is better than the next in every sense, given that there are no other evaluation metrics.

One of the most exciting things is that the recall has a value of 0.871 for nine out of ten algorithms, making the range of recall and f2.0 scores very small. On the other side, both the precision and f0.5 have a larger range of about 0.02.

Based on those two observations, something should be investigated further; however,that will be left for evaluation. (See Section 7.1.5)

Lastly, the LabelSpreading algorithm has the best score in all five metrics, with an f0.5 of 0.971, f2.0 of 0.894, precision of 0.999 and a recall of 0.871, with the subsequent two algorithms having the same score after rounding.

6.6 Hyperparameter tuning

This section details the settings and test results generated by the hyperparameter tuning.

There are two main points of interest. Firstly, the final configuration of all the hyperparameters for each tuned algorithm, and secondly, the test results compare the real-world performance of the algorithms in their base configuration vs the tuned versions.

After 100 training sessions, each algorithm's final configuration is decided by the best validation performance gained in Optuna. Then the test results are presented for the base and tuned version of the algorithms.

6.6.1 Suspicious Conversations Identification

The Table 6.5a shows the configuration of the hyperparameter tuned algorithms for the SCI module. The most important information from this table is that the CalibratedClassifierCV (CCCV) does not take in any parameters because it has an internal routine for hyperparameter tuning.

All of the feature types for all of the results in the test results for hyperparameter tuning of the SCI module are Term Frequency-Inverse Document Frequency (TF-IDF) as seen in Table 6.6a. This is because it has been inherited from Table 6.4a where almost all of the solutions prefered TF-IDF, in effect determining what the feature type would be in the following table.

Classifier	Parameter	Search Space	Final Value
SVC - TF-IDF	c	1e-2 - 1e2	18.86
	gamma	1e-4 - 1e1	0.1018
	kernal	rbf, poly, sigmoid	sigmoid
CalibratedClassifierCV - TF-IDF	NaN	Nan	Nan
MLP - TF-IDF	hidden_layer_size	10-100	(13,40,100)
	activation	tanh	tanh
	alpha	1e-5 - 5e-2	0.000505
	solver	adam	adam
	learning_rate	adaptive	adaptive
Ensemble	SVC_weigth	1 - 10	9.495
	MLP_weigth	1 - 10	5.773
	CCCV_weigth	1 - 10	7.824

(a) Hyperparameter optimization settings for Suspicious Conversation Identification

Classifier	Parameter	Search Space	Final Value
SGD - TF	alpha	1e-8 - 1	0.0009466
	tol	1e-8 - 1	0.0001361
	loss	"modified_huber", "perceptron"	"modified_huber"
	penalty	"l1", "l2", "elasticnet"	"l2"
LR - Bin-BOW	C	1e-4 - 1e4	7.905
	tol	1e-4 - 1e4	158
	penalty	'l2'	l2
	solver	newton-cg,lbfgs,sag,saga	lbfgs
MLP - BOW	hidden_layer_size	10-100	(42,22,10,10)
	activation	tanh	tanh
	alpha	1e-5 - 5e-2	4.750e-04
	solver	adam	adam
	learning_rate	adaptive	adaptive
Ensemble	SGD_weigth	1 - 10	9.453
	MLP_weigth	1 - 10	6.335
	LR_weigth	1 - 10	4.730

(b) Hyperparameter optimization settings for Victim from Predator disclosure

Classifier	Parameter	Search Space	Final Value
RF	n_estimators	50-1000	544
	criterion	gini,entropy	gini
	max_depth	10-200	44
	min_samples_split	2-15	10
	min_samples_leaf	2-10	2
	max_features	sqrt,auto	sqrt
XGB	n_estimators	10-1e3	724
	learning_\rate	1e-3 - 1e1	1.0875
	max_depth	30-1000	924
	subsample	1e-4 - 1	0.9454
KNN	n_neighbours	2-50	22
	metric	manhatten	manhatten
	leaf-size	2-50	4
Ensemble	RF_weigth	1 - 10	0.333
	XGB_weigth	1 - 10	3.147
	KNN_weigth	1 - 10	3.442

(c) Hyperparameter optimization settings for Social Behavioural Biometrics

Table 6.5: Settings for hyperparameter optimization

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
Tuned Soft Voting Ensemble	*TF-IDF*	***0.987***	*0.961*	*0.953*	0.997	*0.997*
Soft Voting Ensemble	*TF-IDF*	*0.987*	*0.961*	*0.953*	0.996	*0.997*
Tuned SVC	*TF-IDF*	0.985	**0.962**	**0.954**	0.994	*0.997*
Base SVC	*TF-IDF*	0.985	0.947	0.935	**0.999**	*0.997*
MLP	*TF-IDF*	*0.985*	*0.950*	*0.939*	*0.997*	*0.997*
Tuned MLP	*TF-IDF*	*0.984*	*0.950*	*0.939*	*0.996*	*0.997*
CCCV	*TF-IDF*	0.984	0.959	0.951	0.993	*0.997*

(a) Test Results for Hyperparameter tuned Suspicious Conversation Identification

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
Tuned Soft Voting Ensemble	Multi-Feature	**0.944**	0.828	0.795	**0.990**	1.000
Base Soft Voting Ensemble	Multi-Feature	0.944	**0.837**	**0.807**	0.986	1.000
Tuned SGD	TF	0.936	0.823	0.791	0.981	1.000
Base MLP	Bin-BOW	0.936	0.804	0.768	**0.990**	1.000
Tuned LR	Bin-BOW	0.931	0.800	0.764	0.985	1.000
Base LR	Bin-BOW	0.931	0.800	0.764	0.985	1.000
Tuned MLP	Bin-BOW	0.929	0.802	0.768	0.980	1.000
Base SGD	TF	0.923	0.814	0.783	0.966	1.000

(b) Test Results for Hyperparameter tuned Victim from Predator disclosure

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
Tuned Soft Voting Ensemble	SBB	**0.935**	**0.829**	**0.799**	**0.976**	*1.000*
Soft Voting Ensemble	SBB	0.928	0.828	**0.799**	0.967	*1.000*
RF	SBB	0.921	0.827	**0.799**	0.958	*1.000*
Tuned RF	SBB	0.911	0.825	**0.799**	0.944	*1.000*
Tuned XGB	SBB	0.911	0.825	**0.799**	0.944	*1.000*
XGB	SBB	0.889	0.820	**0.799**	0.914	*1.000*
KNN	SBB	0.833	0.807	**0.799**	0.842	*1.000*
Tuned KNN	SBB	0.833	0.807	**0.799**	0.842	*1.000*

(c) Test Results for Hyperparameter tuned Social Behavioural Biometrics

Table 6.6: Test results for hyperparameter tuned algorithms

We can see that two out of three algorithm pairs have been positively impacted from being optimised. However, this is slightly optimistic as two out of the three algorithm pairs fluctuate with 0.001 in all metrics between the base and the tuned version.

The pairing of algorithms that do have some impact from being hyperparameter tuned is the Support Vector Machines—incidentally being the algorithm with the highest f2.0 and recall, being slightly better than the following algorithm the soft voting ensembles in both metrics with 0.001.

Lastly, we can see that ensembles perform better for the SCI problem. They are scoring an f0.5 of 0.987 for the tuned soft voting ensembles. Although the base soft voting ensembles the same score for all metrics aside from precision, the tuning has a 0.001 higher score.

6.6.2 Victim from Predator

From the results in Table 6.6b there are several points of interest, but first, one note has to be made about the Feature Type "Multi-Feature". It denotes that each of the classifiers within the voting classifier has distinct feature types, so they cannot be labelled as a single feature type.

As opposed to the SCI module, it is clear that the tuning impacts the VFP. All algorithms perform as good or better after having been tuned. Especially in the case of the SGD classifier, the difference between the tuned and base variations are pretty significant.

For the primary metric, the best performing algorithm is again the tuned ensemble with a 0.944. Naturally, with the best f0.5, the ensemble also has the highest precision with a 0.99. However, this score is shared with the base MLP. The base ensemble achieves the best recall and f2.0, 0.837 and 0.807, respectively.

6.6.3 Social Behavioural Biometrics classifiers

One of the most exciting features of the Table 6.6c is that the table is perfectly sorted, regardless of which column, meaning that the ordering is which solutions are strictly better than the others.

Another feature is that all algorithms have the precisely same recall of 0.799, which is the same that we could see for the validation results in Table 6.4c. Such a pattern means that all performance changes between the algorithm come from higher or lower precision.

There are some mixed results regarding tuning. The RandomForest preferred the base implementation, while the k-nearest neighbors (KNN) did not seem to be impacted by tuning. In comparison, the ensemble and XGBoost prefered the tuned version.

Generally, the fluctuation in precision between the pairs of algorithms is above 0.01. In effect, having some impact, but not a large one, on the performance.

Lastly, the strictly best algorithm was the tuned ensemble, having a score an f0.5 of 0.935, f2.0 of 0.829 with a precision of 0.976 and the 0.799 recall shared with all other solutions.

6.7 Test results

This section presents the test results for the two approaches, which is equivalent to the actual score in the PAN-12 competition.

The results are presented by joining the Social Behavioural Biometrics (SBB) classifiers' results and the complete two-chain classifier score.

Firstly one thing should be reiterated. All the scores for the two-stage classifiers are unique since there are two classifiers connected in series, which means that the first classifier impacts the results of the second classifiers. In effect, errors in the first classifier propagate and compound to the second classifier.

For this reason, all of the results are calculated by using a shared classifier for the SCI part of the problem. All of the VFP classifiers are using the conversations that an MLPClassifier with TF-IDF classified as suspicious, which is the best classifier for SCI as seen by the validation results in Table 6.4a.

When analysing Table 6.7, it is immediately apparent that there is a pattern in the best classifiers. The top five algorithms have the same score and are all SBB classifiers. Aside from these possible anomalies, the highest performing solution is the RidgeClassifierCV, with TF-IDF having an f0.5 of 0.947. The highest precision, 0.990, is obtained by the Tuned Soft Voting Ensemble, which is a score shared with the two BOW-based MLP algorithms. Lastly, the best performing algorithm in the form of recall is the PAC with TF-IDF, which subsequently has the highest F2.0, with scores of 0.835 and 0.86, respectively.

A final note should be made that there is a preference in the test results for the SBB and TF-based features, having twenty-two out of thirty entries.

Model	**Feature Type**	**f0.5**	**f2.0**	**Recall**	**Precision**	***Accuracy***
LabelSpreading	*SBB*	*0.952*	*0.833*	*0.799*	*1.000*	*1.000*
DecisionTreeClassifier	*SBB*	*0.952*	*0.833*	*0.799*	*1.000*	*1.000*
RandomForestClassifier	*SBB*	*0.952*	*0.833*	*0.799*	*1.000*	*1.000*
ExtraTreesClassifier	*SBB*	*0.952*	*0.833*	*0.799*	*1.000*	*1.000*
MLPClassifier	*SBB*	*0.952*	*0.833*	*0.799*	*1.000*	*1.000*
RidgeClassifierCV	TF-IDF	**0.947**	0.848	0.819	0.986	*1.000*
MLPClassifier	TF-IDF	0.946	0.844	0.815	0.986	*1.000*
MLPClassifier	TF	0.946	0.844	0.815	0.986	*1.000*
SGDClassifier	TF-IDF	0.946	0.844	0.815	0.986	*1.000*
PassiveAggressiveClassifier	TF-IDF	0.945	**0.860**	**0.835**	0.977	*1.000*
Tuned Soft Voting Ensemble	Multi-Feature	0.944	0.828	0.795	**0.990**	*1.000*
LinearSVC	TF-IDF	0.944	0.847	0.819	0.981	*1.000*
Base Soft Voting Ensemble	Multi-Feature	0.944	0.837	0.807	0.986	*1.000*
MLPClassifier	BOW	0.941	0.818	0.783	0.990	*1.000*
MLPClassifier	Bin-BOW	0.938	0.811	0.776	0.990	*1.000*
SGDClassifier	TF	0.938	0.830	0.799	0.981	*1.000*
CalibratedClassifierCV	TF-IDF	0.937	0.827	0.795	0.981	*1.000*
RidgeClassifierCV	TF	0.936	0.833	0.803	0.976	*1.000*
Tuned SGD	TF	0.936	0.823	0.791	0.981	*1.000*
LinearSVC	TF	0.936	0.823	0.791	0.981	*1.000*
RidgeClassifier	Combined	0.935	0.839	0.811	0.972	*1.000*
RidgeClassifier	TF-IDF	0.935	0.839	0.811	0.972	*1.000*
XGBClassifier	SBB	0.935	0.829	0.799	0.976	*1.000*
Tuned Soft Voting Ensemble	SBB	0.935	0.829	0.799	0.976	*1.000*
SGDClassifier	Bin-BOW	0.933	0.835	0.807	0.972	*1.000*
PassiveAggressiveClassifier	TF	0.933	0.835	0.807	0.972	*1.000*
Perceptron	TF-IDF	0.932	0.842	0.815	0.967	*1.000*
CalibratedClassifierCV	TF	0.931	0.809	0.776	0.980	*1.000*
Tuned LR	Bin-BOW	0.931	0.800	0.764	0.985	*1.000*
Logistic Regression	Bin-BOW	0.931	0.800	0.764	0.985	*1.000*

Table 6.7: Top thirty test results

7 Evaluation and Discussion

This chapter presents the evaluation of the results and the contrast to work done by other researchers in the field. The chapter also discusses choices made during the planning and experimentation. Reflecting on how these choices contrast against the work and findings of other researchers.

7.1 Evaluation

This section analyses and evaluates the results for the Thesis. The main points of interest are presented, starting with a subsection dedicated to the performance of the proposed solutions. Following this, two subsections, analysing the value of different feature families and algorithm families for solving tasks within Sexual Predator Identification. The fourth subsection looks into the usage of evaluation metrics in the field and how it impacts the proposed solutions. Following this, a subsection dedicated to the literary review finally closes the section by investigating a state of the art paper with debatable results.

7.1.1 Results analysis

Per stage performance

In Sexual Predator Identification, three stages need to be completed to detect predators when applying the architecture proposed by Villatoro-Tello et al. (2012). Firstly, a prefiltering stage aiming to remove the entries that do not contain enough information to be predicted. Following this, a classifier is used to find all conversations that contain predators, known as the Suspicious Conversation Identification (SCI) module. Lastly, a second classifier is used to discern which users within the suspicious conversation are the predator. This stage is known as the Victim from Predator disclosure (VFP).

There are several observations to make regarding a setup like this. Firstly the prefilter sets an upper bound for the best possible recall on the task. Meaning that, given that out of 100 predators, if only 80 passes the prefilter, the highest recall every obtainable will be 0.8, and the highest f0.5 score will become 0.952. Secondly, the SCI module impacts the learning of the VFP module. There are two cases where the SCI module negatively impacts the VFP module. Firstly, with a lower recall in the SCI module, the VFP will lack enough predators to learn what discerns predators from other people. Secondly, with a lower precision in the SCI module, the dataset for the VFP module will become imbalanced.

Looking at the proposed solution as described in Figure 5.2, we can analyse which step removes the most predators from the test set, which would indicate the most significant

Step	Removed Predators	Precentage of total lost predators
Prefilter	27	59
SCI	12	26
VFP	7	15

Table 7.1: Predators removed per stage

constraint to performance, which can be found in Table 7.1.

It is clear that the prefilter removes most predators, followed by the SCI stage.This could indicate that the complexity of the problem decreases throughout the stages. One explanation for this could be that data quality increases as the data go through the stages. However, this would also imply that the quality increase is more significant than the weaknesses introduced by faulty predictions in the SCI module.

From Section 4.1.1 the dataset is known to be both imbalanced and noisy. Incidentally, the two first stages handle each of the problems. The noisy data is cleaned with the prefilter, removing the data that does not hold enough information to be predicted. While the SCI rebalances the dataset by removing all the conversations with only innocent chat participants, it rebalances the dataset heavily. The class balance after the SCI stage is 215 predators and 164 victims for the proposed solution, meaning that the SCI rebalances to the point of having more predators than victims in the test set.

Proposed Solution

The best performing solution seen in Figure 5.2 is the two-stage classifier with a Multi-Layer Perceptron with TF-IDF for the Suspicious Conversation Identification and a RidgeClassifierCV with TF-IDF for the Victim from Predator disclosure.

Regarding the features, it is clear that it means that there is a preference for TF-IDF, for the analysis of why, Section 7.1.3 goes into detail regarding that issue. The best performing solution from the validation results for the SCI module, as seen in the Table 1 in the appendix, is the Multi-Layer Perceptron. Since the SCI is an intermediary step in detecting predators, only the validation results are being used to evaluate what solution performs well.

A possible reason for the combination of MLP and TF-IDF being the best solution for SCI, can be seen in Section 7.1.3 regarding features, and section 7.1.4 regarding algorithms, where MLP and TF-IDF are the highest performings from both categories. The combination of the most adaptable algorithm and the most adaptable feature making the best solution is not surprising. Ideally, the ensemble methods or the hyperparameter tuning should increase the performance, but as described in Section 7.1.4 the results have been mixed for both apporaches.

The best performing result uses the RidgeClassifierCV with TF-IDF. One of the main properties of RidgeClassifiers is attempting to lower the multicollinearity or oversimplified redundancy of information within features. In other words, the RidgeClassifiers perform well when the features that are strong predictors for the problem are closely related. High

multicollinearity can come from either class, meaning a high degree of multicollinearity between the predators, victims, or both groups. One very likely source of multicollinearity for Sexual Predator Identification is that the textual features, to a certain extent, can represent the emotional, social and psychological differences between predators and other people. Several different words can represent the same psychological differences and, therefore, be highly correlated if this is the case. Given that this is the case, RidgeClassifier seemingly filters the information appropriately.

However, it might be that the preference for the algorithms is a representation of the algorithmic complexity issues outlined in Section 7.1.4. In short, complex algorithms on simple features like TF-IDF can see past the issues that problems like multicollinearity introduce. A good match between algorithm complexity and features being the cause of the high performance is slightly supported by RidgeClassifier and MLP being two of the best algorithms overall, regardless of the module.

7.1.2 Validation and test sets

Another exciting note is the many disconnects between the validation and test scores, as seen in the score for the Suspicious Conversation Identification, found in validation Table 1 and test Table 5 in the appendix. Generally, the recall is higher across all modules in the validation scores, while the precision is higher in the test scores. The most plausible reason for this is the lack of homogeneity in the training and test datasets. The two datasets can be similar from a statistical standpoin. However, this does not ensure that the actual entries within each dataset are similar. For example, with 50 predators in each dataset with an equal amount of messages sent each, the contents for the messages can differ. Some predators are more inclined towards relationship forming, while others are more sexual, meaning that there will be differences between training and test sets.

The lower recall in the test results means more predators evade detection in the test set. In other words, the algorithms are generally more hesitant to predict that a conversation is predatory. This could be due to a larger variety of textual expressions for the predators in the test set than the predators found in the training sets. On another note, the validation sets are one-tenth of a sample with 136 predators, which is lower than the test set with 227 predators. Given that there are significantly more predatory entries in the test set than in the training set, both in absolute values and proportionally, it makes sense that the predators in the test set are more heterogeneous than those in the validation set, simply by being a larger and broader group.

The higher recall and lower precision of the validation score mean that the algorithms seem to be overly eager to predict an entry as a predator during the validation stage. It could point to the predators within the training set being more self-similar, meaning that they exhibit some unique traits. However, these traits are shared with some other chat participants. Since the patterns between the validation and test scores are found for both the Suspicious Conversation Identification (SCI) and the Victim from Predator disclosure (VFP), it is reasonable to assume that the similarity might come from the other participants in the same chat as the predators. In other words, the participants in predatory chats might be more self-similar in the training set than their counterparts in

the test set.

7.1.3 Features

The test results from Table5 in the appendix and validation results from Table 1 in the appendix show a clear preference for TF-based features in the SCI module. Most of the top ten solutions are either TF-IDF or TF features. Given that the field generally has tiny margins for results, it is challenging to discern concrete patterns that can explain the reasons for the preference of these techniques. Nonetheless, several possible explanations come to mind.

Firstly, TF-based features perform better for compound scores, which means that the average score of TF-based solutions is higher than for other features. However, the highest score reachable is shared with BOW for the recall and precision. An example can be seen from the best test result in Table 5. The best solution is an SVC with Bin-BOW. The more granular metrics give some context, the best test result for recall can be found with the SGDClassifier, which uses Bin-BOW, having a score of 0.96. An interesting pattern can be seen with precision, where the best score for a recall above 0.5 is a 0.999 achieved by PassiveAggressiveClassifier, RidgeClassifier and SVC with TF-IDF. The SVC with BOW-BIN matches that precision.

One of the main reasons for the increase in the compound scores for TF-based solutions is the slightly higher recall that TF-based solutions have compared to the BOW-based solutions. An example of this can be seen in the Victim from Predator disclosure (VFP) test results found in Table 6. Generally, the recall among the top-performing BOW-based solutions is lower than for the top-performing TF-based solutions, and the difference is 0.03. Seemingly this suggests that the general advantage of TF-based features is connected to finding more predators.

There are several possible reasons for the recall being higher with TF-based features. Firstly, BOW-based solutions use the presences and frequency of specific words to determine the contents of a text. In comparison, TF-based features only see the commonality of the words in the document, either locally or in a collection of documents. One possible reason can be that the count of terms that indicate attempts at grooming is not as good as looking at how many rare terms are used. Groomers are different from other people, meaning that it is expected that their language is skewed differently from the rest of the population. However, it might be that predators do not share which words they use, as much as sharing the pattern of using more rare words.

Another reason could be that the distribution of common and rare language within a document is a more powerful predictor than the words used. Oftentimes groomers will mimic the writing of their intended target, which means that they often use simplistic language to seem more approachable for children while also using uncommon terms for the more grooming-related parts of the conversations. Patterns in the commonality of terms could allow algorithms to recognise some of the more elusive predators, as this might be a better marker than the usage of specific words. The slight increase in general performance for TF-based features is most likely connected to the predators not sharing vocabulary as much as patterns in usage of common and rare words or the balance

between the two.

Throughout all the results, one thing sticks out: the consistent lack of Combined and Linguistic Inquiry and Word Count (LIWC) solutions being in tables. Making it clear that LIWC-based features perform subpar at best. There are several possible reasons for this. One thing that is known is that LIWC models less information than BOW-based and TF-based solutions, meaning that it is expected that there is some information loss, making it a weaker feature when used alone. The concern for the lack of ability to model the problem as a whole is the reason for combining the features of LIWC and TF-IDF to a combined feature. The lower performance might be expected for the SCI module, which is more general, trying to find predatory content. However, the VFP module has the explicit goal of discerning victims from predators, where the psychological differences between the groups are essential in discerning the two groups. One of the possible reasons for the lack of performance for the VFP module might be connected to the fact that LIWC is a general feature for personality, psychology and social behaviour. It might not be specific enough to target the traits of the predators, needing more advanced psychological analysis, more akin to looking for dark triad traits within the text. Another possible reason is that using all terms in LIWC might have been unfortunate. Some of the terms in LIWC might be more valuable than others, ending up introducing noise by not removing the less valuable terms from the dictionary.

7.1.4 Algorithms

Classifiers are used in Sexual Predator Identification for two primary purposes: finding conversations containing predators and discerning which chat participants are the predators. With six features across two modules, validation and testing each solution, every single algorithm has been trained 24 times. To better analyse and process the performance of the algorithms, the top five results from the 24 situations is aggregated to a table. The resulting Table 7.2 shows the sum of instances that were preferred by all solutions.

Two notes need to be made about the table. Firstly, the variation in classifiers per family makes the ordering inaccurate. The varying size of each family favoured large families, meaning that the neural_network and the calibration families are the two top performers and should be moved to the top. Aside from the varying family sizes, the ranking is consistent.

Secondly, since the table has been generated based on the classifier families assigned by Sklearn, the python package that implements all the algorithms, the linear_models and Svm families have been split. All SVMs are linear_models, but this is not reflected in Sklearn.

In short linear models, calibration models, and neural networks adapt the best to the problem. One possible reason the linear models perform well is that the problem is linearly separable with the given feature representation and noise removal with the prefilter. Meaning that the complexity of the problem is not too high, favouring easier to compute and understand patterns. Low complexity could be an enticing thought, given that the early testing of more advanced neural networks did not give any considerably

Family	Sum	Name	Instances
Linear_model	49	SGDClassifier	11
		LogisticRegression	13
		RidgeClassifierCV	9
		RidgeClassifier	7
		PassiveAggressiveClassifier	5
		Perceptron	4
Neural_network	21	MLPClassifier	21
Svm	18	LinearSVC	12
		SVC	5
		NuSVC	1
Ensemble	11	AdaBoostClassifier	3
		RandomForestClassifier	5
		ExtraTreesClassifier	3
		StackingClassifier	0
		BaggingClassifier	0
Calibration	10	CalibratedClassifierCV	10
Tree	3	ExtraTreeClassifier	3
		DecisionTreeClassifier	0
Naive_bayes	3	GaussianNB	0
		CategoricalNB	0
		MultinomialNB	3
		BernoulliNB	0
Discriminant_analysis	1	QuadraticDiscriminantAnalysis	0
		LinearDiscriminantAnalysis	1
Dummy	0	DummyClassifier	0
Neighbors	0	NearestCentroid	0
		KNeighborsClassifier	0
Semi_supervised	0	LabelPropagation	0
		LabelSpreading	0

Table 7.2: Usage of algorithms among top solutions

better results. Low complexity should favour the Naivë_Bayes family of classifiers, which has only three instances. Given the weak performance of the Naivë_Bayes classifiers and the complexity of the task, it is reasonable to assume that there is some more advanced information in the features.

This could be due to the problem being highly complex, meaning that the current features are not sufficient to model the information in the dataset correctly. If this assumption is valid, these insufficient features might still contain traces of the more complex information. Generating a situation where the simple models can only interpret the easily accessible patterns in the data. At the same time, the halfways complex models, like the random forest, ends up trying to model more complex relations, but the algorithms are not advanced enough to do so, leaving the simple neural network the only algorithm that can leverage the extra information, giving it a slight edge over the other algorithms.

Highly complex data does not explain why the more advanced neural networks did not work. It would be contradictory to believe the weaker classifiers cannot leverage the information when the advanced algorithms perform worse. One possible reason is that the more advanced features used for the neural networks are more information-dense but not enough to map the complex relations in the data. Leaving the extra information more confusing to the advanced neural network than for a simpler algorithm. The issue then is the relation between the complexity level of features and the strength of the algorithms. Without a more comprehensive selection of features and a wider selection of advanced algorithms, it is not possible to trace the true source of the observations.

The Deep Learning (DL) architecture employed by Liu et al. (2017) was quite similar to the early testing framework from this Thesis. They attempted to use the Long Short-Term Memory (LSTM) version of a Recurrent Neural Network (RNN). This Thesis attempted to use Recurrent Neural Networks and Convolutional Neural Networks as algorithms for the SCI and VFP module. However, this was only attempted with GloVe 500d features and not advanced sentence embeddings as in the case of Liu et al., which supports the notion that there was some mismatch between the findings from this work and their findings.

Disparities in the complexity of the features and models could also explain why the transfer learning did so well because it generates features of an appropriate complexity level rather than taking in other features. However, even though machine learning, deep learning, and transfer learning models show some indications that the relation between features and models is sensitive in SPI, there are some unresolved issues.

Firstly, there is a mismatch between the linear_models, the ensembles and the neural_-network. The relationship between the three is that we have neural_network, ensembles and linear_models in order of descending complexity. For this reason, we should expect that for a set of equally complex features, the three families should rank in the same order. However, this is not the case. The ensembles have lower performance than the linear_models, yet the Neural Network outperforms all. Seemingly there are two possible sources for the results. Firstly, the neural network is adaptable enough to filter away the relations in the data it cannot use and adapt better to several levels of complexity within

features. Secondly, the ensemble families are somehow underperforming in relation to the features used. A possible explanation is that both the ensemble families and the neural network would perform better with more complex features, but the adaptability of neural networks has allowed it to perform better than all other solutions despite not being peak performance. However, this cannot be verified without testing the ensembles and the neural_networks further.

In summary, the different levels of complexity for the algorithms are essential in determining how well a pairing of features and algorithms can perform in Sexual Predator Identification, with TF-IDF as the best-performing feature as described in Section 7.1.3, machine learning algorithms with an average level of complexity perform the best.

Hyperparameter tuning

Hyperparameter tuning was performed with the top three solutions from each module to gain even more performance from the promising solutions. Table 6.6 with the test results for the hyperparameter tuned solutions shows that in five out of seven cases for the two-chain classifier, the tuned variations of the algorithms outperformed the base variation. This would indicate that hyperparameter tuning increases the performance. However, this is slightly deceptive, as there are only two cases where the difference between the tuned and base variations are considerable. Both are in the Victim From Predator (VFP) module seen in Table 6.6b, where the SGD became considerably better, most notably in precision where the tuned version has an increase of 0.15. Contrasting this, the MLP became considerably worse when tuned, but only in precision where the score decreased by 0.1.

The results raise two concerns: why is there a lack of impact for most hyperparameter-tuned solutions, and why has SGD and MLP been impacted?

The lack of impact could be due to a failure in the tuning process. Among the most common issues for hyperparameter tuning is improperly defined search spaces, stopping the Optuna framework from converging on an optimal search space region. This can become an issue if the search space is too large or the framework is given too few training sessions to converge. An issue with a failure in training being the cause for the lack of impact is that wrong tuning should result in some impact, in most cases degraded performance. Lack of impact could only happen if none of the tuned parameters impacts the algorithms.

A more likely reason for the lack of impact is that many different configurations are near the optimal performance for several classifiers. This means that even though there might be a single optimal solution, it will not be significantly better than most other attempts. Even though the tuning is working, the result tables will not show any indications of change in the performance.

The second issue is the large disparities in performance for the tuned Stocastich Gradient Decent Classifier and Multi-layer Perceptron classifier in the Victim from Predator disclosure, seen in the Table 6.6b. Firstly, the SGD classifier had an improved performance of 0.008 for recall and 0.015 for precision. One reasonable explanation for the increase in performance is that the base settings for the SGD are not suboptimal for

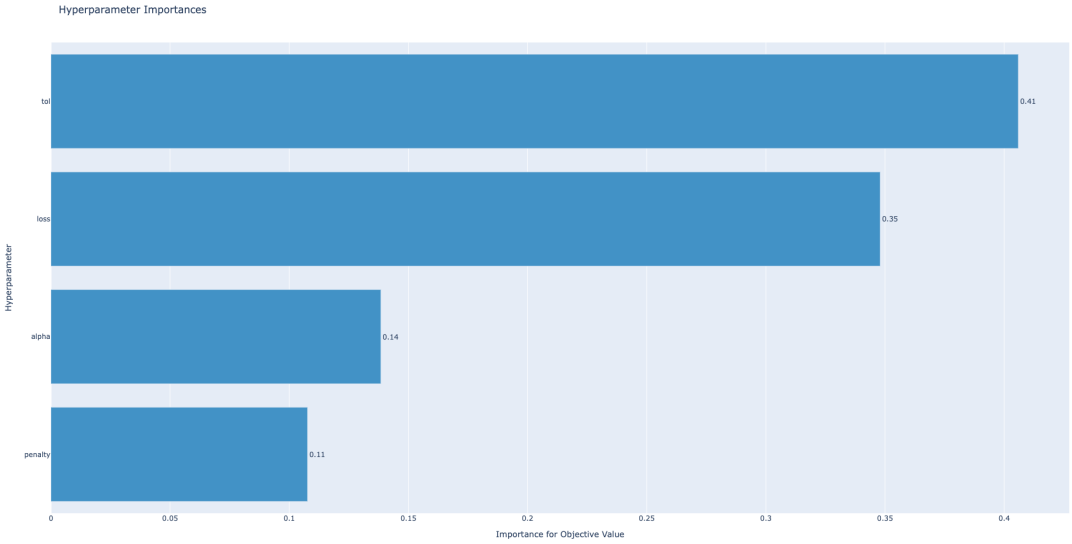


Figure 7.1: Ranking of importance of parameters during hyperparameter tuning for Stochastic Gradient Decent

the problem, possibly quite wrong, giving the optimisation framework more opportunities to improve the final result.

From the Figure 7.1 it is clear that the most impactful parameters are the loss and tolerance, so evaluting the final configuration seen in Table 6.5 against the standard found in the documentation from Sklearn [1] should give some insights.

The hyperparameter importance figure might give a wrong impression due to the search space for the loss only being two values, "modified_huber" and "perceptron", which neither are the default for the classifier. By default, the classifier uses "hinge", which gives a linear SVM, which was excluded favouring greater variation for the ensemble between the LR and the SGD. The changed perspective on the loss makes tolerance even more important. For an easier analysis it has been generated a Figure 7.2 which shows the the top trainings for the SGD classifier, and what parameters were favoured by the training framework. Seeing that there is a concentration on the right hand column, between 0.001 and 0.001 means that a tolerance that is set slighly below the default of 0.001 is preferable and is most likely the source of the increased performance from the tuning.

There are some likely sources regarding the lowered performance from the Multi-Layer Perceptron. The most common issue would be a faulty search space for this algorithm; however, as seen in Table 6.5 the MLP classifiers are tuned both for the VFP and SCI stages and with the same search spaces, which means that comparing the validation results for the hyperparameter tuning found in Table 4 should give some insights. However, the lowered performance is only present in the VFP stage, suggesting that the issue is not

[1] SciKitLearn

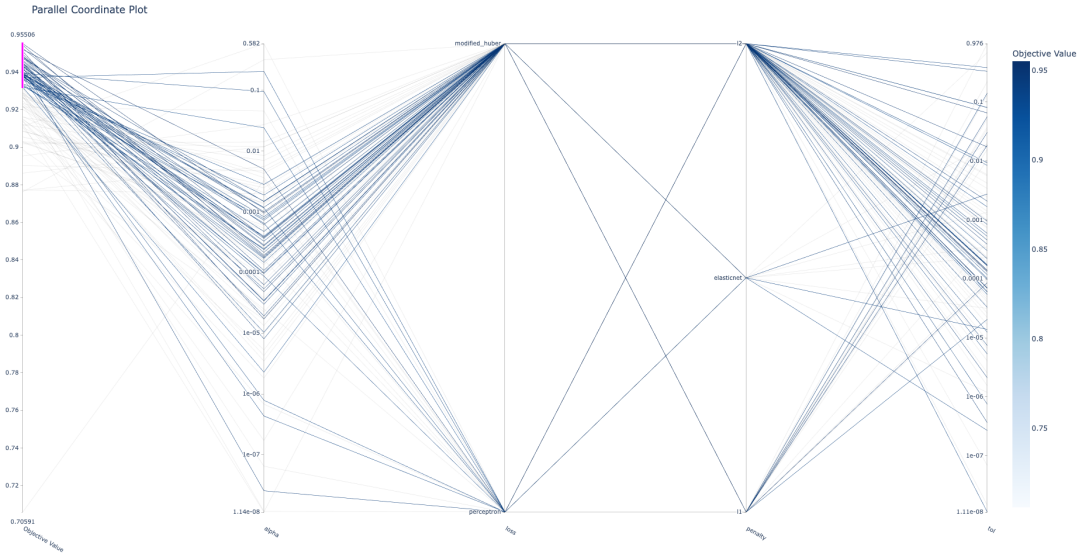


Figure 7.2: Hyperparameter tuning training sessions for Stochastic Gradient Decent

related to the search spaces.

Other possible reasons for a lowered performance could be connected to the differences in the problems. As mentioned in the Section 7.1.1 the complexity of the problems decreases for the later stages of the task, which means that the VFP stage should be easier to solve than the SCI stage. An easy problem can be closer to optimal performance without being tuned, which would suggest that tuning on a simple problem should yield less impact. Conversely, this harms the tuning.

A more likely reason would then be that the differences between the training and test set are more impactful for the VFP stage than for the SCI stage. In other words, attempting to tune the problem towards the training set should lower the performance; however, neither the test results from Table 6.6 nor the validation results from Table 4 support this.

For the sake of completeness, an issue regarding the Logistic Regression for the Victim from Predator disclosure (VFP) module seen in Table 6.5b has to be mentioned. A misconfiguration in the code allowed the algorithm to have a vast search space for the tolerance (tol). Ideally, this tolerance would never go above 1, but due to a misconfiguration, the tolerance and C shared search spaces even though they are very different parameters. The misconfiguration uncovered an interesting fact that the tolerance was set to 158, which is an absurdly large number in many senses.

Meaning that for the Logistic Regression, a parameter had been set to something unusable. The most likely reason is that the parameter is not important, or at least less important than other parameters for Logistic Regression Looking at the available graphs from Optuna containing information about the training sessions, we can see that this is the case in Figure 7.3.

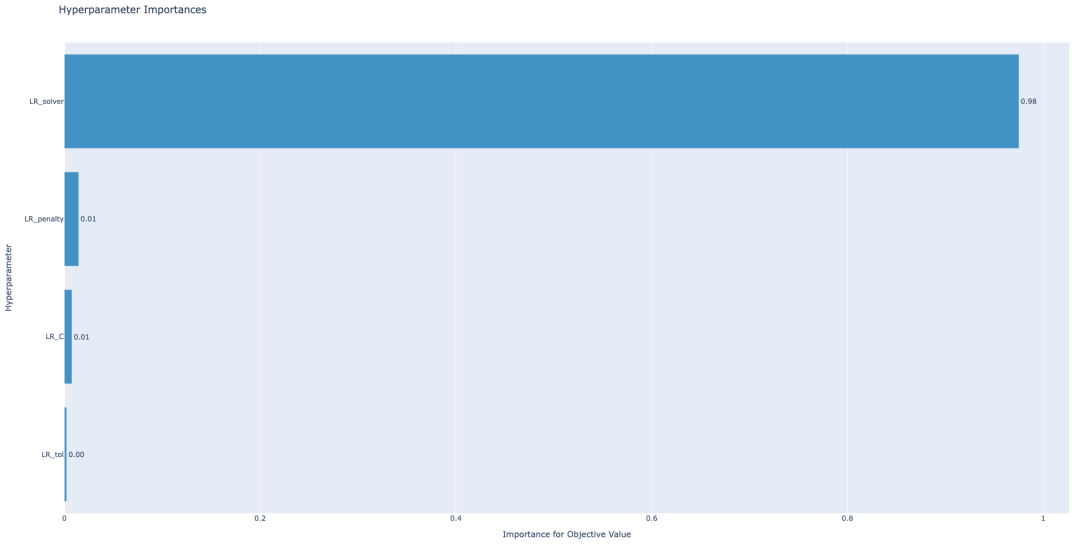


Figure 7.3: Ranking of importance of parameters during hyperparameter tuning for Logistic Regression

Ensembles

One attempt to boost the proposed solutions' predictive power was combining the top three algorithms in each module into soft voting ensembles.

Looking at the test results in Table 6.6 the ensembles perform the best for all modules. In the Susipicous Conversation Detection (SCI) module, the margins between the solutions were small, meaning that there are no only slight indiciations in regards to the cause of the increased performance. A slight increase like this might come from the strongest classifier, in this case, the SVC, carrying most of the weight but the increased recall from the CalibratedClassifierCV being able to sway the vote in cases where the SVC could not discern the pattern. For the Victim from Predator disclosure (VFP) module seen in Table 6.5b, it is clear the score improvement comes from a higher recall, mostly having a recall 0.02 points above the average from the other solutions. Given the high test results, it is an indication that the ensembles can leverage the strengths of the different classifiers within the ensembles, seeing as the recall for the base and tuned ensembles are higher than any of the individual subclassifier in the respective ensembles.

However, even though the tuned soft voting ensemble performs the best in test scores among the hyperparameter tuned algorithms, it does not mean that it has the highest test scores. By comparing the values in the completed SCI test score from Table 5 in the appendix, it is clear that the best solution has an f0.5 of 0.989 while the tuned classifier is in second place with 0.987. This disparity most likely stems from the fact that the soft voting ensembles usage of confidence values from the algorithms disallows several of the best algorithms from being used within the ensembles as mentioned in the Sections 6.3.1 and 6.3.2.

For the validation results for the Victim from Predator disclosure (VFP), as seen in Table 4b, the ensembles perform almost 0.015 below the best single classifier solution, and in addition, only two out of eight solutions are below the two soft voting classifiers. One of the prerequisites for bagging classifiers performing well is having a medium to high variance in the models. If they do not, the strongest classifier will, in effect, be confused by the votes from the weaker classifiers, introducing noise. The cause of the lowered performance only being in the VFP module might be due to the Stochastic Gradient Descent (SGD) classifier and the Logistic Regression (LR) classifier being from the same family.

One of the main findings is the differences in score between the validation results as seen in Table 4 and the test results seen in Table 6.6. The most likely cause for the differing scores comes from the heterogeneity of the training and test set as explored in Section 7.1.2.

Hyperparameter tuning is more focused and involves fewer algorithms making inconsistencies in scores between runs more visible. The changed scores can be seen in the hyperparameter tuning validation Table 4c in the appendix and VFP validation results from Table 2. The clearest example of this Multi-Layer Perceptron (MLP) with Bin-BOW. During the training done to select algorithms for tuning, it gained a recall of 0.956 and a precision of 0.958, while when the base implementation was trained during the hyperparameter tuning, it had a recall of 0.927 and a precision of 0.964. In Sexual Predator Identification, a difference of 0.03 in recall is relatively large. Interestingly this also made it such that in the validation results from Table 2 the MLP was the best performing algorithm and Logistic Regression with Bin-Bow being the second best, however when the algorithms were trained for the hyperparameter tuning seen in Table 4c the ordering flipped.

One of the possible reasons for changes in results between runs could be the random seed. The random seed is a number that is used to calculate random values, which most algorithms use at some point during training. The random seed is usually set in advance to get consistent results between training. The variables are not isolated without doing this, meaning it is impossible to conclude what changes have impacted performance. Interestingly, the Logistic Regression (LR) implementation of ScikitLearn (Sklearn) sets the global random seed[2], both overriding the one set by a user if they do that and, therefore, controlling the behaviour of the succeeding algorithms within the same file. For the MLP classifier, which in this case has switched place with the LR between the initial validation training and the hyperparameter tuning base run, there is only one way to override this unintended behaviour, by setting the random_state parameter for the algorithm. Incidentally, this is the case for this Thesis, as the LazyPredict package sets random seeds within itself by using a defined random_state for all classifiers that have the parameter. The LR is run before the other classifiers in code the hyperparameter tuning, meaning that two different locked random seeds have been used between the validation and base hyperparameter tuning runs.

A second and often more plausible reason for varying validation scores between runs of

[2] https://github.com/scikit-learn/scikit-learn/issues/15266

the algorithm would be that the random seed used to generate the cross-validation of the datasets is changing. However, this was set in advance, independently of each run, meaning that the dataset used to train the algorithms have not been impacted by the varying random seeds.

7.1.5 Social Behavioural Biometrics

An interesting note has about the Social Behavioural Biometrics (SBB) classifier approach to Sexual Predator Identification is that the publication Wani et al. (2021) had some inaccuracies. Due to the irregularities found in the initial stages of trying to recreate the Predatory and Victim Words Vocabulary (PreVicVocab) as described in Section 6.4.1, more time was invested in figuring out the cause of these irregularities.

The first finding was that some extensive oversampling of the training set had been done. One of the examples is that one of the predatory authors[3] has 48 entries in the training set and one entry in the test set. In the case of a two-stage classifier, the same author can be present in many conversations, but since the SBB classifier only uses the authorships rather than the conversations, there cannot be duplicate authors.

Further analysis of the data frame shows there are 233 groups of duplicated entries, all groups having the size of 48 entries. From the prefiltering stage that is shared between two-stage classifiers and SBB, the number of predators after filtering should be 234, so it is feasible to assume that this is the source of this number. In effect, this means that the paper does not describe all the actions done to the data in advance of the training, making it incomplete.

The keen-eyed reader might have caught the reference that there was an overlapping entry across the training and test sets. The intersection was 77% in the test set and 100% in training. An intersection grade translates to how tainted a dataset is, which means every predator was encountered 48 times during training before being encountered in the test set. A tainted dataset outright invalidates all the test results presented, meaning that they cannot be compared to other solutions.

In short, this is the basis for the adjustments made to the results table in the preceding Chapter 6 excluding the SBB classifier as the best performing solution, since this would make the use of the results without further investigation disingenuous. Most of these findings can be due to mishaps or incompleted work. However, it makes it such that a new study needs to be performed to verify the approach's merits as a solution for SPI. Nothing indicates that the SBB features are not well suited for SPI. However, the paper that presented the approach and presentation is insufficient to claim the results or the validity of the feature presented.

7.2 Discussion

This section will discuss how well the Thesis has been able to answer the research objective and research questions by looking at the work done, comparing the findings to other

[3] (4a9332d7466b98d11c23e4447b26460a)

work, and analysing if the justifications for choices made during the development of the system still are valid, in light of the findings and evaluation.

Research question 1 *What approaches are the most promising for detecting predators?*

In order to get a proper understanding of what can be considered a promising approach to detecting predators, a literary review was conducted using Structured Literary Review (SLR), a method to ensure that the most relevant publications are found.

However, as mentioned in Section 3.1, a small multidisciplinary field where many of the authors have different research backgrounds does not lend itself as well to Structured Literary Review. The main reason is that the language each author use to set titles will be significantly impacted by which field the author considers their main field of study.

Snowballing and reverse searching were employed to handle the challenges posed by search engines' lower than desired performance. In short, leveraging authoritative papers as anchoring points to find most of the related work, as described in Section 3.2.2. In effect, this introduced a trade-off between quantity and quality, where the SLR will, in the end, find almost all the good papers, even though the effort to do so will be higher, snowballing and revers searching is less structured, especially when combined; however, it will give higher quality results in a shorter amount of time.

The trade-off seems reasonable given the inability to leverage the usually powerful functionalities of research search engines to any good effect. It is impossible to compare the choice against other researchers as currently, it has not been encountered any high-quality, in-depth discussions or evaluations from other researchers of how to handle the complexity of naming in the field.

For the contents of the literary review, there are several trends in the best performing detection systems for predators. Generally, it can be seen that there are some overlapping things between most of the most successful approaches:

- They all acknowledge the low data quality by using a filtering method to remove noise from the datasets they use.
- Most researchers split whichever task they want to solve into subproblems, most aptly by using the SCI and VFP modules, but also for other approaches.
- There have been some overarching movements towards understanding predators' behavioural and emotional aspects, which have continuously shown much promise.
- Highly advanced features can perform well, but only with appropriately advanced models.

However, most researchers encountered during the literary review had chosen some feature or algorithm that was a core part of the approach they sought to explore. Due to the inherent complexity of the task, which is the root cause for the two-stage classifier architecture proposed by Villatoro-Tello et al. (2012), coupled with the fact that there are many ways of presenting results, and most implementations of the two-stage classifier

has some slight configurations that are different, like prefilter settings, which makes it hard to compare the findings between research directly.

For this reason, it was chosen to focus more of the efforts on lightweight approaches in such a sense that they could easily be implemented alongside each other. This means that some nodes containing promising results might not have been picked as nodes for further snowballing and reverse searching, meaning that there might be gaps in the literary review for some families of features and algorithms considered more advanced. It is pretty unlikely that this has had any tangible impact, as snowballing had been performed from the key publications in the field like Inches and Crestani (2012) and Pendar (2007) which every single other notable publication has referenced.

In short, even though there are patterns in the approaches that perform well, which will constitute what can be considered promising results, due to issues with reporting, and incomparable architectures, it is hard to give more than indications about what can be considered promising approaches, for this reason, it was deemed better to proceed with a cautious approach, which did not assume too much about what would give increased performance.

Research question 2 *What types of features are viable and optimal for use in detecting predators?*

Research Question 2 seeks to use the insights gained from Research Question 1 to decide a set of features that were considered promising that could be used for further investigation to find which of the investigated features are the most viable.

Initially, based on the findings from research question 1, it was clear that there existed several approaches that had hand-crafted features or features that somehow were tightly coupled to either the algorithm or the approach used to detect predators. The inflexible nature of these features was deemed unsatisfactory, meaning behavioural features, lexical chains of sexual terms, and several deep learning implementations like advanced sentence embeddings were initially considered as less desirable.

Even though they were considered undesirable, behavioural features based on the description from Cardei and Rebedea, word embeddings in the form of GloVe 500d were implemented; however they were developed before the full selection of algorithms were ready, so they were only tested with Multinomial Naive Bayes, SVMs and Logistic Regression. The results were lacklustre, as mentioned in Section 6.3.3 regarding the early testing that was attempted.

For this reason, Bag-of-words (BOW), Binary bag-of-words (Bin-BOW), Term-Frequency (TF), Term Frequency-Inverse Document Frequency (TF-IDF), Linguistic Inquiry and Word Count (LIWC) and Combined(TF-IDF+LIWC) were chosen as the features to explore, in addition to Moodbook and a specialised variation of BOW for the Social Behavioural Classifier. All these features share the core trait that they are able to be automatically generated consistently across several sources and can run with any machine learning system.

The consequences of the choice to use textual features can maybe be most clearly seen in the section discussing the highest performing solutions from Section 7.1.1 and the

evaluation of the algorithms in Section 7.1.4. Since the selected algorithms are quite simple features, it seems to have impacted the more advanced algorithms from deep learning that were tested during the early stages to perform worse. That conclusion is drawn mostly based on the fact that many researchers listcite have been able to gain comparable scores to the state of the art with deep learning and matching features.

The findings can be seen in the Section 7.1.3 from evaluation, where generally, as a family, TF-based features performed the best, even having the interesting trait of generally having slightly higher compound scores, mostly due to an overall increase in recall over the other feature families.

The findings stand in contrast to most findings, where each researcher seems to find their own preferred features for each stages of the two-stage classifier. The reason for the disparity can come from several sources and is most likely connected to one of the limitations for many publications in the field, being the fact that few of the publications share enough of the implementation to easily compare the findings of others.

However, given that the f0.5 scores from the top-ten solutions for both the Suspicious Conversation Identification and the Victim from Predator disclosure are so high, it is feasible to conclude that the TF-based features are will perform better than BOW-based solutions as a rule of thumb; however, some BOW-based solutions can slightly outperform the best TF-IDF solutions. While LIWC has had a subpar performance overall.

Research question 3 *What categories of algorithms are most suited for use to detect predators?*

Research Question 3 seeks to use the insights gained from Research Question 1 to decide which algorithms should be used, in the same manner as Research Question 2 has done.

Many different algorithms have been employed based on the available literature, most notably Neural Networks (NN) and Support Vector Machines (SVM). However, as no studies have used enough algorithms to compare scores directly between algorithms, most researchers seemingly use either SVM, NN or a small selection of other algorithms. However, to avoid the problem altogether, an approach inspired by Fauzi and Bours (2020) was employed, where a large selection of algorithms were used. There is a crucial difference between this Thesis and the work by Fauzi and Bours which is that they selected the algorithms themselves, while the proposed solutions take the largest selection of available algorithms that are compatible with the approach within reason.

An immediate limitation of this approach is, therefore, that heavier algorithms from Deep Learning and Transfer Learning are not easily integrated into the same system, meaning that they, for this reason, has not been explored in-depth, making the comparison of which algorithms perform well, only applicable to traditional machine learning algorithms.

Besides the publication from Fauzi and Bours there are not any other works using large enough selections of algorithms and features to make it possible to compare the findings from Section 7.1.4 with. Generally, it is a trend towards linear models, basic neural networks and calibration models performing well. The findings would suggest

Rank	Researcher	Precision	Recall	f0.5
1	Cardei and Rebedea (2017)	1.000	0.818	0.957
2	Liu, Suen, and Ormandjieva (2017)	1.000	0.811	0.955
3	**Ulberg**	0.986	0.819	0.947
4	Fauzi and Bours (2020)	0.956	0.858	0.934
5	Villatoro-Tello, Juárez-González, Escalante, Montes-y-Gómez, and Pineda (2012)	0.9804	0.7874	0.9346

Table 7.3: Comparing our results to other solutions in the field

that medium complexity and highly complex machine learning algorithms perform well. However, the findings are highly connected to the set of features as described in Section 7.1.3, meaning that the findings are only valid for simple features.

Goal *Determine if a quantitative approach to algorithm design performs well for SPI*

With the research questions presented in Section 1.2 and the evaluation and discussion for the questions, the goal of the Thesis has been reached. Training on data presented as a part of the PAN-12, a large selection of twenty-nine algorithms and six features across two modules for a total of 348 training scenarios has been developed. A quantitative approach to developing solutions to detect predators has successfully been developed.

From the experiments, it is clear that within the bounds of the combinations of algorithms and features, TF-based features and neural networks are preferred, however generally showing small margins between the top-performing solutions. Comparing the solutions against other researchers in the field as seen in Table 7.3, the proposed solution currently has the third-highest score.

State of the field Taking into account the performance seen in the solutions in the field today, coupled with the test results achieved by our approach, it is reasonable to question what problem is the right one to solve within Sexual Predator Identification. A famous concept, the "Theory of Constraints", states that every process has a constraint that is the largest, and total throughput can only be increased by alleviating the constraint.

In terms of real-world benefits, today's limiting factor is not the algorithmic solutions' performance, as discussed in the paragraph in Section 7.2 regarding the quality of solutions. The main constrain is the lack of access to realistic data. It is stated as one of the main concerns by Pendar (2007), the first author in the field, flagging it as a significant issue.

As seen in the comparison of solutions in the field, the two top performers, which incidentally has the highest performance on one module each, are both publications from 2017. This suggests that much work has been given out since then, not acknowledging that combining the strength of the approaches could have generated something close to an optimal solution several years ago.

The current solutions will be good at detecting possible abuse on platforms where it is feasible to deploy extensive large-scale analysis of all messages. However, the performance is only guaranteed for one-to-one messaging and generally under other conditions like the assumption that adults pretending to be children will be sufficient to generate realistic grooming content.

Given that Internet culture is fast-moving and children are early adopters of new technologies, predators will follow suit. In effect, it makes it such that data will be outdated regarding iconography, slang, and platform-specific lingo or actions. Getting high-quality data to make SPI solutions is a continuous task rather than a one-time effort like the PAN-12 dataset.

The lack of papers pursuing this well-known data issue is probably layered. One of the possible barriers to entry is that collecting the data is complex, so using the PAN-12 dataset is common since it reduces complexity for all studies that do not have data collection as the primary goal. Secondly, given that it is possible to collect relevant data, getting the desired data from real predatory situations would be preferred. Given that these two already significant hurdles are overcome, a researcher would still need permission to share the dataset publicly.

In short, this Thesis does not address the most pressing issue, and neither does the main body of work within the field; however, this should not detract from the importance of the new contributions. Finding new approaches to detect predators is valuable; however, the direct results from each approach might be changed if new data becomes available. The main contribution is finding promising approaches, while the specific implementations and results might be less interesting.

Quality of solutions in the field Due to the inherent complexity of the field, reading and interpreting the results is not simple. One subtly is that even though we have f0.5, recall and precision, the dataset PAN-12 is noisy, and it is not known if it is even possible to find all the predators in the test set, even with a perfect solution. For this reason, the results in the field should be contextualised a bit. 27 of the 254 predators are removed from the test set with the current prefiltering settings. As mentioned in the Section 7.1.1 the prefilter is still the stage that removes most of the predators. Lowering the number of predators that get removed while still removing all the data that can be considered noise should be one of the best ways of increasing performance on the PAN-12 dataset. However, retaining more predators requires some way of evaluating the performance of the prefilter, which does not currently exist. The closest approximation would be using the subtask in early detection, that is, to find the lower bound of information needed to detect a predator. As this is not trivial, it is left as a suggestion for future work to investigate how to optimise the prefilter and what insights can be gained from that.

Without a way to evaluate the performance of the prefilter, it can be assumed for the sake of discussion that it is in a close to optimal state already, which would allow for some further analysis of the results in the field. An optimal prefilter gives a limit for recall, meaning that the highest obtainable recall in this scenario is 0.893. However, such a limit does not indicate how well the current approaches in the field perform or how well the proposed solution works.

Combining the best solution to the Suspicious Conversation Identification (SCI) and Victim From Predator disclosure (VFP) modules, scored on recalls, allows us to approximate how good the approaches in the field are. Using the Deep Learning SCI module from Liu et al. (2017) and the VFP proposed by Cardei and Rebedea (2017), as seen in

Module	Method	Researcher	Precision	Recall	f0.5
SCI	Two layer LSTM (Sentence Vectors)	Liu	0.996	0.998	0.996
VFP	Cost Balanced Random Forest (Behavioural Features)	Cardei	0.993	1	0.994
SPI	Merged best solutions	Theoretical	0.993	0.892	0.971
SPI	Best possible solution	Theoretical	1.0	0.893	0.977

Table 7.4: Best currently known solution per stage[4]

Table 7.4[4] would yield the best result possible. Merging the two approaches gets a total recall of 0.919 after applying the adjustments to the scores necessitated by the prefilter removing predators.

With the practical limit to the recall, the corresponding limit for the f0.5 score would be 0.977, with the merged solution having an f0.5 of 0.971. Meaning 99.39% of the practically available max performance is already reachable. Making the same adjustments for the full approach from Cardei and Rebedea would translate to 97.95% of the max performance and 96.93% for the proposed solution. In short, it indicates that the algorithmic solutions to the problem are already very advanced, given the constraints, and that an approximation of the best solutions that can be obtained with current methods is already almost performing perfectly.

It is essential to be careful not to overstate the value of insights like this since there are several challenges to making too many assumptions based on the information presented here. Firstly, the practical limit only shows how well the current solutions perform within the currently known boundaries. It can, at best, give indications of the upper bound of performance for detecting predators within this dataset. Secondly, a known issue is that solving problems in stages introduces error propagation, the fact that low precision or recall in the SCI module will make the training set for the VFP incorrect. Since the VFP module is trained on what it assumes to be truthful information from the SCI module, any form of errors will worsen the VFP module. The issue is further compounded by the fact that differing prefiltering settings between approaches make it such that the basis that each classifier is trained on is different, even if the evaluation is the same for all solutions. In effect, it is impossible to plainly add the scores from the different approaches to get a score for the best currently available solution. The merged solution is an approximation of the best possible outcome, meaning that combining the two approaches most likely will degrade the performance.

An analysis like this only helps contextualise the results from the current solutions in the field, and the proposed solution, showing that evaluating the results of approaches in the field of Sexual Predator Identification has several subtleties that should be considered.

[4]Neither the SCI nor VFP module has recall adjusted for the predators removed by the prefilter.

8 Conclusion and Future Work

Online grooming attempts are an ongoing risk for all children's wellbeing, especially as more youths access the Internet and phones that allow them to interact with strangers. The work in the field of Sexual Predator Identification shows that it is possible to detect predators from online chat logs, which is an essential step in moving away from manually finding groomers to larger automatic systems. The Thesis seeks to use a quantitative approach to drive decisions with data. A general framework using lightweight features and algorithms has been proposed to improve the ability to find predators. The framework allowed several hundred combinations of features and algorithms to be tested, enabling the comparison of results to find the underlying reasons for the performance of features and algorithms. The study managed to find highly performant combinations for each subtask within the problem of detecting predators. The results show many viable combinations, meaning that focusing on maximal performance in the primary metric might not be desirable; instead, it should be prefered to look at a combination of runtime, implementational complexity and performance. The chapter will describe the contributions made to the field of Sexual Predator Identification. Closing the final chapter of the Thesis with suggestions for future work, which would be valuable additions to the field.

8.1 Contributions

There are three main contributions from the Thesis: Firstly, the usage of a quantitative approach to designing algorithmic solutions within SPI, secondly, the review of the related work and analysis of the current state of the field, and lastly, the invalidation of the results in the publication presenting the Social Behavioural Biometrics approach to Sexual Predator Identification (Wani et al., 2021).

The main contribution is a flexible framework that allows us to expand with more algorithms and features to test approaches to find the optimal combination given the available resources. The framework can be explained as a step-wise optimisation of the following:

- Feature choice
- Algorithm choice
- Combining algorithms to ensembles
- Hyperparameter tuning

The resulting framework aided in finding the optimal solution given the available twenty-nine algorithms, and six features, spread across two modules was an MLPClassifier with TF-IDF to find suspicious conversations and a RidgeClassifierCV with TF-IDF for the victim from predator disclosure, resulting in a total score of 0.947, placing it as the best approach from the PAN-12 competition but making it the third given the current state of the field. The value of working within a framework like this will only increase over time as more features, algorithms and steps of the process are rigorously tested from a quantitative approach. The value of this contribution is heavily tied to the second contribution, which was the review of related literature. To my knowledge, it has not been noted how well the best solutions perform and that the prefilter is the main bottleneck for performance on the PAN-12 dataset. In effect, this makes any attempts at increasing performance on the PAN-12 dataset redundant. However, this does not devalue the approaches being developed on the PAN-12 dataset, only those approaches with the primary goal of improving performance on the PAN-12 dataset. With this in mind, the contribution of the framework shows that removing intuition and using data to drive decisions in experimentation for SPI is a feasible approach.

Lastly, the final contribution is reviewing and invalidating the results for the Social Behavioural Biometrics approach proposed by Wani et al.. In short, based on the description of the architecture proposed, it would not be possible to gain a recall of 0.96 as claimed, due to the prefilter setting an upper bound for recall at 0.917. It is not mentioned that the training set was oversampled, in addition to the test and training sets having a big intersection, where all predators in the test set were already present in the training set, which means that the test-set is not a hold-out invalidating any results generated from the dataset.

8.2 Future Work

This section will present other considered approaches, methods or techniques that could be applied to improve the current approaches or other thoughts about how to approach work in the field discovered during the project.

8.2.1 Automatic exploration of unknown datasources

As per the paper by Europol (2020), they were able to find grooming data by using LIWC's sexual terms and FastText, a word embedding, to find the misspellings and related terms. By searching within the sexual terms and using topic modelling, they could discern grooming in large scale unlabelled data sources.

However, sexuality is not the only discriminatory part of grooming behaviour. An example is that predators spend more time forming relationships with the children than talking about sexual themes (Gupta et al., 2012). In addition, there are several other ways than purely sexual ways to identify the discerning characteristics of predators as explored in Section 3.3.5 concerning nature describing features.

Using emotion features to explore many datasets should find clusters of predators,

which can be manually reviewed for quality. The addition of new sources of predators would allow for better true positives, and hopefully, find ways of identifying good sources for false positives and false negatives. In short, laying the groundwork for building more robust, generalised algorithms that are more agnostic to new platforms, the evolution of Internet slang, and the new iconography that are invented and put into use over time as platforms mature.

A subtlety has to be mentioned. The generation of the PAN-12 dataset as described in Section 4.1.1 was considered a one-time task rather than building a system that could be continuously used to generate new data. The primary goal should be to build a framework or approach to continuously making new sources of predatory content accessible to researchers, limiting the issues of grooming evolving, and changing as new platforms and textual methods of expression appear.

8.2.2 Prefilter settings

As mentioned in the evaluation, in Sections 7.1.1 and 7.2 the current bottleneck for most approaches that follow the style of Villatoro-Tello et al. (2012) is the prefilter. Work to introduce the prefilter as a step in the framework described in Section 8.1 would allow for finding better settings. Several approaches can be attempted to find better settings, and they all will share the need for some statistical analysis of the dataset to find some theoretical underpinning for what in chat logs can be considered to hold information—either being amount of messages, the total length of content for an author or conversation, or completely different metrics. Even though early detection has the same goal of finding the lower bound of information needed to detect predators, the prefilter is at the same time different in nature. The prefilter seeks to find the balancing point of removing irrelevant entries and keeping predators, while early detection is more concerned with keeping all predators. However, the shared goal might make early detection an excellent field to explore possible ways of evaluating lower bounds of information needed to detect predators.

8.2.3 Combining solutions from other researchers

As demonstrated in Section 7.2 under the paragraph regarding the quality of the solutions in the field, there already exists two or more solutions that are close to the maximum performance we can gain with the PAN-12 dataset using the two-stage classifier, setting aside the issue of prefilters described in Section 7.1.1. However, it is also impossible to know how the error propagation between the stages will impact each module, meaning that merging two approaches might not get the theoretical max from adding the numbers. With this uncertainty, it would be best to have a study that can conclude whether or not it is possible to merge approaches and obtain the practical maximum performance given the restrictions in recall imposed by the prefilter on detecting predators with the PAN-12 dataset.

8.2.4 AutoML

Incidentally, the framework in this approach is curiously similar to another field of study known as AutoML. AutoML is a field that tries to automate every step of machine learning, describing it as a way to input any data and get out an optimal solution. Given an extension of the current AutoML applications that would allow for the usage of two-classifiers and more extensive feature sets that are specifically meant for SPI, it would lower the implementation time for each new approach within the field.

Bibliography

Susanne Baumgartner, Patti Valkenburg, and Jochen Peter. Assessing causality in the relationship between adolescents' risky sexual online behavior and their perceptions of this behavior. *Journal of youth and adolescence*, 39:1226–39, February 2010. doi: 10.1007/s10964-010-9512-y. URL https://pubmed.ncbi.nlm.nih.gov/20177962/.

Dasha Bogdanova, Paolo Rosso, and Thamar Solorio. On the impact of sentiment and emotion based features in detecting online sexual predators. In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, pages 110–118, Jeju, Korea, July 2012a. Association for Computational Linguistics. URL https://aclanthology.org/W12-3717.

Dasha Bogdanova, Paolo Rosso, and Thamar Solorio. Modelling fixated discourse in chats with cyberpedophiles. In *Proceedings of the Workshop on Computational Approaches to Deception Detection*, pages 86–90, Avignon, France, April 2012b. Association for Computational Linguistics. URL https://aclanthology.org/W12-0413.

Dasha Bogdanova, Paolo Rosso, and Thamar Solorio. Exploring high-level features for detecting cyberpedophilia. *Computer Speech & Language*, 28(1):108–120, January 2014. doi: 10.1016/j.csl.2013.04.007. URL https://doi.org/10.1016/j.csl.2013.04.007.

Amparo Elizabeth Cano, Miriam Fernández, and Harith Alani. Detecting child grooming behaviour patterns on social media. In Luca Maria Aiello and Daniel A. McFarland, editors, *Social Informatics - 6th International Conference, SocInfo 2014, Barcelona, Spain, November 11-13, 2014. Proceedings*, volume 8851 of *Lecture Notes in Computer Science*, pages 412–427. Springer, November 2014. doi: 10.1007/978-3-319-13734-6_30. URL https://doi.org/10.1007/978-3-319-13734-6_30.

Claudia Cardei and Traian Rebedea. Detecting sexual predators in chats using behavioral features and imbalanced learning. *Natural Language Engineering*, 23(4):589–616, January 2017. doi: 10.1017/S1351324916000395. URL https://doi.org/10.1017/S1351324916000395.

Yun-Gyung Cheong, Elin Rut Gudnadottir, Alaina K. Jensen, Julian Togelius, Byung Chull Bae, and Christoffer Holmgård Pedersen. Detecting predatory behaviour in online game chats. In *The 2nd Workshop on Games and NLP : Workshop at the 6th International Conference on Interactive Digital Storytelling, GAMNLP-13 ; Conference date: 09-11-2013 Through 09-11-2013*, November 2013. URL http://gamesandnarrative.net/icids2013/call-to-participate-in-workshops.

Mohammadreza Ebrahimi, C. Suen, and O. Ormandjieva. Detecting predatory conversations in social media by deep convolutional neural networks. *Digital Investigation*, 18:33–49, July 2016. doi: https://doi.org/10.1016/j.diin.2016.07.001. URL https://www.sciencedirect.com/science/article/abs/pii/S1742287616300731.

April Edwards, Lynne Edwards, Jen Bayzick, India McGhee, Amanda Leatherman, and Kristina Moore. Comparison of rule-based to human analysis of chat logs. Presented at 1st International Workshop on Modelling Social Media (MSM09). Seville, Spain. Nov 2009, November 2009. URL https://www.researchgate.net/publication/239929144_Comparison_of_Rule-based_to_Human_Analysis_of_Chat_Logs.

Hugo Jair Escalante, Manuel Montes y Gomez, Luis Villasenor, and Marcelo Luis Errecalde. Early text classification: a naïve solution. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 91–99, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-0416. URL https://aclanthology.org/W16-0416.

Europol. Exploiting isolation: Offenders and victims of online child sexual abuse during the covid-19 pandemic. Technical report, Europol, June 2020. URL https://www.europol.europa.eu/publications-events/publications/exploiting-isolation-offenders-and-victims-of-online-child-sexual-abuse-during-covid-19-pandemic.

Muhammad Ali Fauzi and Patrick Bours. Ensemble method for sexual predators identification in online chats. In *8th International Workshop on Biometrics and Forensics*, pages 1–6. IEEE, April 2020. doi: 10.1109/IWBF49977.2020.9107945. URL https://doi.org/10.1109/IWBF49977.2020.9107945.

Cathy Girouard. Ncmec. Technical report, US Department of Justice, Office of Justice Programs, Office of Juvenile Justice and Delinquency Prevention, July 2008. URL https://www.ojp.gov/pdffiles1/ojjdp/fs200128.pdf.

Aditi Gupta, Ponnurangam Kumaraguru, and Ashish Sureka. Characterizing pedophile conversations on the internet using online grooming. *CoRR*, abs/1208.4324, August 2012. URL http://arxiv.org/abs/1208.4324.

Chad M. Harms. Grooming: An operational definition and coding scheme. *Sex Offender Law Report*, 8(01):1–6, January 2007. URL https://www.civicresearchinstitute.com/online/article.php?pid=7&iid=250.

José María Gómez Hidalgo and Andrés Alfonso Caurcel Díaz. Combining predation heuristics and chat-like features in sexual predator identification. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Rome, Italy, September 17-20, 2012*, volume 1178 of *CEUR Workshop Proceedings*. CEUR-WS.org, September 2012. URL http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-HidalgoEt2012.pdf.

Giacomo Inches and Fabio Crestani. Overview of the international sexual predator identification competition at PAN-2012. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Rome, Italy, September 17-20, 2012*, volume 1178 of *CEUR Workshop Proceedings*. CEUR-WS.org, September 2012. URL http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-InchesEt2012.pdf.

Mohammadreza Keyvanpour, Necmiye Genc Nayebi, Mohammadreza Ebrahimi, Olga Ormandjieva, and Ching Y. Suen. *Automated Identification of Child Abuse in Chat Rooms by Using Data Mining*, pages 245–275. IGI Global, June 2016. ISBN 9781522504641. doi: 10.4018/978-1-5225-0463-4.ch009. URL https://www.igi-global.com/chapter/automated-identification-of-child-abuse-in-chat-rooms-by-using-data-mining/157462.

Sylvia Kierkegaard. Cybering, online grooming and ageplay. *Computer Law & Security Review*, 24:41–55, March 2008. doi: 10.1016/j.clsr.2007.11.004. URL https://www.sciencedirect.com/science/article/abs/pii/S0267364907001082.

Jinhwa Kim, Yoon Jo Kim, Mitra Behzadi, and Ian G. Harris. Analysis of online conversations to detect cyberpredators using recurrent neural networks. In Archna Bhatia and Samira Shaikh, editors, *Proceedings for the First International Workshop on Social Threats in Online Conversations: Understanding and Management*, pages 15–20. European Language Resources Association, May 2020. URL https://aclanthology.org/2020.stoc-1.3/.

April Kontostathis and Amanda Leatherman. Chatcoder: Toward the tracking and categorization of internet predators. In *PROC. TEXT MINING WORKSHOP 2009 HELD IN CONJUNCTION WITH THE NINTH SIAM INTERNATIONAL CONFERENCE ON DATA MINING (SDM 2009). SPARKS, NV. MAY 2009*, January 2009. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.151.6501.

Halvor Kulsrud. Detection of cyber grooming in online conversation. Master's thesis, Norwegian University of Science and Technology, June 2019. URL https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2623143.

Matthieu Latapy, Clémence Magnien, and Raphaël Fournier. Quantifying paedophile queries in a large P2P system. In *INFOCOM 2011. 30th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 10-15 April 2011, Shanghai, China*, pages 401–405. IEEE, April 2011. doi: 10.1109/INFCOM.2011.5935191. URL https://doi.org/10.1109/INFCOM.2011.5935191.

Dan Liu, Ching Yee Suen, and Olga Ormandjieva. A novel way of identifying cyber predators. *CoRR*, abs/1712.03903, November 2017. URL http://arxiv.org/abs/1712.03903.

H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, April 1958. doi: 10.1147/rd.22.0159. URL https://ieeexplore.ieee.org/document/5392672.

Nikolaos Lykousas and Constantinos Patsakis. Large-scale analysis of grooming in modern social networks. *CoRR*, abs/2004.08205, April 2020. URL https://arxiv.org/abs/2004.08205.

India Mcghee, Jennifer Bayzick, April Edwards, Lynne Edwards, Alexandra Mcbride, and Emma Jakubowski. Learning to identify internet sexual predation. *International Journal of Electronic Commerce*, 15(3):103–122, April 2011. doi: 10.2307/41300734. URL https://www.tandfonline.com/toc/mjec20/15/3.

Saif Mohammad and Peter Turney. Emotions evoked by common words and phrases: Using Mechanical Turk to create an emotion lexicon. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 26–34, Los Angeles, CA, June 2010. Association for Computational Linguistics. URL https://aclanthology.org/W10-0204.

Colin Morris and Graeme Hirst. Identifying sexual predators by SVM classification with lexical and behavioral features. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Rome, Italy, September 17-20, 2012*, volume 1178 of *CEUR Workshop Proceedings*. CEUR-WS.org, September 2012. URL http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-MorrisEt2012.pdf.

Suraiya Jabin Mudasir Ahmad Wani, Nancy Agarwal and Syed Zeeshan Hussain. User emotion analysis in conflicting versus non-conflicting regions using online social networks. *Telematics and Informatics*, 35(8):2326–2336, November 2018. ISSN 0736-5853. doi: https://doi.org/10.1016/j.tele.2018.09.012. URL https://www.sciencedirect.com/science/article/pii/S0736585318308402.

Loreen Olson, Joy Daggs, Barbara Ellevold, and Teddy Rogers. Entrapping the innocent: Toward a theory of child sexual predators' luring communication. *Communication Theory*, 17:231 – 251, July 2007. doi: 10.1111/j.1468-2885.2007.00294.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1468-2885.2007.00294.x.

Rachel O'Connell. A typology of child cybersexploitation and online grooming practices. Technical report, Cyberspace Research Unit, University of Central Lancashire, July 2003. URL http://image.guardian.co.uk/sys-files/Society/documents/2003/07/24/Netpaedoreport.pdf. https://www.theguardian.com/society/2003/jul/17/childrensservices.technology.

Javier Parapar, David E. Losada, and Alvaro Barreiro. A learning-based approach for the identification of sexual predators in chat logs. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF 2012 Evaluation Labs and Workshop, Online*

Working Notes, Rome, Italy, September 17-20, 2012, volume 1178 of *CEUR Workshop Proceedings*. CEUR-WS.org, September 2012. URL http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-ParaparEt2012.pdf.

Nick Pendar. Toward spotting the pedophile telling victim from predator in text chats. In *Proceedings of the First IEEE International Conference on Semantic Computing (ICSC 2007), September 17-19, 2007, Irvine, California, USA*, pages 235–241. IEEE Computer Society, September 2007. doi: 10.1109/ICSC.2007.32. URL https://doi.org/10.1109/ICSC.2007.32.

James W. Pennebaker, Ryan L. Boyd, Kayla Nicole Jordan, and Kate G. Blackburn. *The Development and Psychometric Properties of LIWC2015*. University of Texas at Austin, September 2015. doi: 10.15781/T29G6Z. URL https://repositories.lib.utexas.edu/handle/2152/31333.

Robert Plutchik. A general psychoevolutionary theory of emotion. In *Theories of Emotion*, volume 1, pages 3–33. Academic Press, January 1980. doi: https://doi.org/10.1016/B978-0-12-558701-3.50007-7. URL https://www.sciencedirect.com/science/article/pii/B9780125587013500077.

Hady Pranoto, Fergyanto Gunawan, and Benfano Soewito. Logistic models for classifying online grooming conversation. *Procedia Computer Science*, 59:357–365, December 2015. doi: 10.1016/j.procs.2015.07.536. URL https://www.researchgate.net/publication/283042589_Logistic_Models_for_Classifying_Online_Grooming_Conversation.

Karen Spärck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21, January 1972. URL https://www.emerald.com/insight/content/doi/10.1108/eb026526/full/html.

Esaú Villatoro-Tello, Antonio Juárez-González, Hugo Jair Escalante, Manuel Montes-y-Gómez, and Luis Villaseñor Pineda. A two-step approach for effective detection of misbehaving users in chats. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Rome, Italy, September 17-20, 2012*, volume 1178 of *CEUR Workshop Proceedings*. CEUR-WS.org, September 2012. URL http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-VillatoroTelloEt2012b.pdf.

Matthias Vogt, Ulf Leser, and Alan Akbik. Early detection of sexual predators in chats. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 4985–4999. Association for Computational Linguistics, August 2021. doi: 10.18653/v1/2021.acl-long.386. URL https://doi.org/10.18653/v1/2021.acl-long.386.

Mudasir Ahmad Wani, Nancy Agarwal, and Patrick Bours. Sexual-predator detection system based on social behavior biometric (ssb) features. *Procedia Computer Science*,

189:116–127, July 2021. URL https://www.sciencedirect.com/science/article/pii/S1877050921011704.

Melissa Wollis. Online predation: A linguistic analysis of online predator grooming. Master's thesis, College of Agriculture and Life Sciences, Social Sciences of Cornell University, June 2011. URL https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.966.9198&rep=rep1&type=pdf.

Appendices

1 Validation Results

Table 1: Full table of SCI validation result

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
Base MLP	TF-IDF	0.978	0.974	0.973	0.979	0.997
MLPClassifier	TF-IDF	0.978	0.973	0.972	0.979	0.997
Soft Voting Ensemble	TF-IDF	0.977	0.977	0.978	0.977	0.997
Tuned MLP	TF IDF	0.977	0.976	0.975	0.978	0.997
Tuned SVM	TF-IDF	0.976	0.978	0.979	0.976	0.997
Tuned Soft Voting Ensemble	TF-IDF	0.976	0.978	0.979	0.976	0.997
CalibratedClassifierCV	TF-IDF	0.975	0.975	0.975	0.976	0.997
LinearSVC	TF-IDF	0.975	0.972	0.970	0.977	0.997
Base CCCV	TF-IDF	0.975	0.975	0.975	0.975	0.997
PassiveAggressiveClassifier	TF-IDF	0.975	0.976	0.977	0.974	0.997
SGDClassifier	TF-IDF	0.974	0.971	0.970	0.975	0.996
MLPClassifier	TF	0.972	0.963	0.960	0.975	0.996
CalibratedClassifierCV	TF	0.971	0.966	0.964	0.973	0.996
Base SVC	TF-IDF	0.971	0.950	0.943	0.979	0.995
SVC	TF-IDF	0.971	0.950	0.943	0.979	0.995
SVC	TF	0.970	0.958	0.954	0.974	0.995
MLPClassifier	Bin-BOW	0.970	0.958	0.954	0.974	0.995
LinearSVC	TF	0.969	0.966	0.965	0.971	0.996
RidgeClassifierCV	TF-IDF	0.966	0.930	0.918	0.979	0.993
RidgeClassifier	TF-IDF	0.966	0.930	0.918	0.979	0.993
RidgeClassifierCV	TF	0.966	0.948	0.942	0.972	0.994
SGDClassifier	TF	0.966	0.965	0.965	0.966	0.995
LogisticRegression	Bin-BOW	0.965	0.952	0.948	0.970	0.995
CalibratedClassifierCV	Bin-BOW	0.965	0.955	0.952	0.969	0.995
SVC	Bin-BOW	0.965	0.941	0.933	0.973	0.994
MLPClassifier	BOW	0.965	0.960	0.958	0.967	0.995
Perceptron	TF-IDF	0.964	0.959	0.957	0.967	0.995
RidgeClassifier	Combined	0.964	0.937	0.928	0.973	0.994
LinearSVC	Bin-BOW	0.964	0.958	0.957	0.965	0.995
RidgeClassifier	TF	0.962	0.928	0.917	0.974	0.993
XGBClassifier	TF-IDF	0.959	0.934	0.926	0.968	0.993
XGBClassifier	Bin-BOW	0.959	0.937	0.929	0.967	0.993
PassiveAggressiveClassifier	TF	0.959	0.969	0.973	0.955	0.995
RidgeClassifierCV	Combined	0.958	0.913	0.900	0.974	0.992
XGBClassifier	Combined	0.957	0.920	0.908	0.970	0.992
SGDClassifier	Bin-BOW	0.953	0.967	0.972	0.949	0.995
XGBClassifier	BOW	0.953	0.925	0.916	0.963	0.992
PassiveAggressiveClassifier	Bin-BOW	0.952	0.936	0.931	0.957	0.993
LogisticRegression	BOW	0.951	0.933	0.927	0.957	0.992
XGBClassifier	TF	0.950	0.911	0.898	0.964	0.991
RidgeClassifierCV	Bin-BOW	0.950	0.891	0.873	0.971	0.990
AdaBoostClassifier	Combined	0.945	0.939	0.937	0.947	0.992
LogisticRegression	TF	0.944	0.871	0.849	0.971	0.988
LinearSVC	BOW	0.943	0.936	0.934	0.945	0.992
Perceptron	Bin-BOW	0.943	0.948	0.949	0.941	0.993
LogisticRegression	TF-IDF	0.939	0.839	0.811	0.978	0.986
MLPClassifier	Combined	0.938	0.947	0.950	0.935	0.992
AdaBoostClassifier	TF	0.935	0.920	0.915	0.941	0.991
Perceptron	TF	0.934	0.929	0.928	0.938	0.991
SGDClassifier	BOW	0.933	0.949	0.954	0.927	0.992
AdaBoostClassifier	BOW	0.932	0.909	0.902	0.940	0.990
AdaBoostClassifier	TF-IDF	0.931	0.914	0.908	0.937	0.990
AdaBoostClassifier	Bin-BOW	0.930	0.912	0.906	0.937	0.990
Perceptron	BOW	0.920	0.936	0.942	0.914	0.990
PassiveAggressiveClassifier	BOW	0.917	0.921	0.922	0.916	0.989
CalibratedClassifierCV	BOW	0.916	0.783	0.748	0.971	0.982
RidgeClassifier	Bin-BOW	0.908	0.878	0.869	0.919	0.986
BaggingClassifier	Combined	0.887	0.794	0.767	0.923	0.980

Table 1 continued from previous page

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
BaggingClassifier	TF-IDF	0.883	0.768	0.736	0.931	0.979
BaggingClassifier	BOW	0.882	0.802	0.778	0.914	0.981
BaggingClassifier	Bin-BOW	0.880	0.803	0.780	0.909	0.980
CalibratedClassifierCV	Combined	0.878	0.716	0.676	0.952	0.976
LabelSpreading	TF-IDF	0.871	0.937	0.962	0.851	0.986
BaggingClassifier	TF	0.871	0.763	0.733	0.914	0.978
LabelPropagation	TF-IDF	0.867	0.938	0.964	0.846	0.986
LinearSVC	Combined	0.862	0.844	0.844	0.874	0.980
KNeighborsClassifier	BOW	0.851	0.696	0.657	0.919	0.974
MLPClassifier	LIWC	0.848	0.810	0.801	0.866	0.978
XGBClassifier	LIWC	0.847	0.759	0.734	0.881	0.976
LogisticRegression	Combined	0.840	0.761	0.739	0.870	0.975
LogisticRegression	LIWC	0.832	0.757	0.735	0.862	0.975
RidgeClassifierCV	BOW	0.817	0.773	0.760	0.835	0.973
ExtraTreesClassifier	Bin-BOW	0.812	0.544	0.490	0.973	0.966
DecisionTreeClassifier	Combined	0.806	0.792	0.788	0.811	0.974
DecisionTreeClassifier	BOW	0.801	0.795	0.793	0.803	0.974
RandomForestClassifier	LIWC	0.801	0.580	0.531	0.917	0.966
CalibratedClassifierCV	LIWC	0.800	0.585	0.537	0.912	0.966
DecisionTreeClassifier	Bin-BOW	0.800	0.794	0.792	0.802	0.973
LabelSpreading	TF	0.790	0.871	0.902	0.766	0.975
ExtraTreesClassifier	LIWC	0.788	0.543	0.493	0.928	0.964
ExtraTreesClassifier	BOW	0.786	0.499	0.446	0.972	0.963
AdaBoostClassifier	LIWC	0.782	0.708	0.687	0.810	0.969
LabelPropagation	TF	0.781	0.873	0.910	0.754	0.974
DecisionTreeClassifier	TF	0.778	0.774	0.773	0.779	0.971
DecisionTreeClassifier	TF-IDF	0.775	0.757	0.751	0.781	0.970
RandomForestClassifier	Combined	0.775	0.478	0.424	0.977	0.962
RandomForestClassifier	BOW	0.771	0.472	0.418	0.977	0.961
RandomForestClassifier	Bin-BOW	0.770	0.474	0.421	0.974	0.961
BaggingClassifier	LIWC	0.768	0.591	0.549	0.853	0.964
LinearDiscriminantAnalysis	LIWC	0.759	0.560	0.515	0.863	0.963
NearestCentroid	TF-IDF	0.757	0.890	0.946	0.721	0.972
ExtraTreesClassifier	Combined	0.757	0.450	0.396	0.982	0.960
RandomForestClassifier	TF-IDF	0.730	0.413	0.361	0.980	0.958
KNeighborsClassifier	LIWC	0.727	0.642	0.618	0.762	0.962
KNeighborsClassifier	Combined	0.727	0.642	0.618	0.762	0.962
SVC	BOW	0.725	0.447	0.401	0.962	0.960
MultinomialNB	Bin-BOW	0.718	0.785	0.811	0.699	0.965
ExtraTreesClassifier	TF-IDF	0.707	0.388	0.338	0.977	0.956
RandomForestClassifier	TF	0.692	0.369	0.319	0.978	0.955
KNeighborsClassifier	Bin-BOW	0.692	0.514	0.474	0.784	0.957
SGDClassifier	LIWC	0.687	0.710	0.724	0.683	0.958
RidgeClassifier	BOW	0.686	0.738	0.757	0.671	0.960
SGDClassifier	Combined	0.685	0.722	0.739	0.677	0.959
KNeighborsClassifier	TF-IDF	0.685	0.886	0.983	0.637	0.962
PassiveAggressiveClassifier	Combined	0.676	0.572	0.552	0.765	0.959
BernoulliNB	Combined	0.668	0.401	0.354	0.859	0.954
ExtraTreesClassifier	TF	0.644	0.321	0.275	0.976	0.952
SVC	LIWC	0.642	0.355	0.311	0.899	0.952
PassiveAggressiveClassifier	LIWC	0.642	0.598	0.595	0.672	0.952
SVC	Combined	0.640	0.353	0.308	0.900	0.952
BernoulliNB	Bin-BOW	0.624	0.358	0.313	0.831	0.951
BernoulliNB	BOW	0.624	0.358	0.313	0.831	0.951
BernoulliNB	TF-IDF	0.624	0.358	0.313	0.831	0.951
BernoulliNB	TF	0.624	0.358	0.313	0.831	0.951
KNeighborsClassifier	TF	0.618	0.848	0.968	0.567	0.949
NearestCentroid	Bin-BOW	0.606	0.763	0.835	0.567	0.947
RidgeClassifier	LIWC	0.602	0.298	0.255	0.926	0.950
GaussianNB	TF	0.599	0.391	0.350	0.729	0.949
MultinomialNB	BOW	0.590	0.848	0.994	0.536	0.943
GaussianNB	TF-IDF	0.584	0.388	0.349	0.703	0.948
Perceptron	LIWC	0.582	0.704	0.778	0.555	0.936
DecisionTreeClassifier	LIWC	0.567	0.572	0.574	0.566	0.943
Perceptron	Combined	0.562	0.674	0.783	0.542	0.909
RidgeClassifierCV	LIWC	0.550	0.269	0.230	0.898	0.948
GaussianNB	Bin-BOW	0.540	0.338	0.301	0.678	0.945
ExtraTreeClassifier	TF-IDF	0.539	0.476	0.458	0.565	0.941
ExtraTreeClassifier	Combined	0.537	0.497	0.485	0.552	0.940
NearestCentroid	TF	0.521	0.797	0.968	0.467	0.925
LinearSVC	LIWC	0.521	0.587	0.759	0.571	0.829
ExtraTreeClassifier	TF	0.515	0.474	0.463	0.531	0.938
ExtraTreeClassifier	BOW	0.494	0.481	0.478	0.499	0.934
ExtraTreeClassifier	Bin-BOW	0.482	0.461	0.454	0.490	0.933
NearestCentroid	BOW	0.477	0.692	0.821	0.433	0.915
GaussianNB	BOW	0.443	0.559	0.644	0.438	0.910
ExtraTreeClassifier	LIWC	0.438	0.445	0.448	0.436	0.925

Table 1 continued from previous page

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
BernoulliNB	LIWC	0.181	0.412	0.714	0.153	0.720
GaussianNB	LIWC	0.172	0.345	0.916	0.155	0.348
MultinomialNB	Combined	0.172	0.313	0.431	0.149	0.801
MultinomialNB	LIWC	0.158	0.425	0.970	0.131	0.574
NearestCentroid	Combined	0.150	0.320	0.512	0.128	0.738
NearestCentroid	LIWC	0.150	0.320	0.512	0.128	0.738
LinearDiscriminantAnalysis	Bin-BOW	0.122	0.302	0.598	0.102	0.625
QuadraticDiscriminantAnalysis	LIWC	0.114	0.329	0.893	0.094	0.425
LinearDiscriminantAnalysis	Combined	0.112	0.286	0.598	0.093	0.584
LinearDiscriminantAnalysis	BOW	0.101	0.262	0.556	0.084	0.571
LinearDiscriminantAnalysis	TF	0.098	0.258	0.568	0.081	0.550
GaussianNB	Combined	0.097	0.299	1.000	0.079	0.213
LinearDiscriminantAnalysis	TF-IDF	0.089	0.241	0.553	0.074	0.513
QuadraticDiscriminantAnalysis	BOW	0.081	0.246	0.838	0.066	0.210
QuadraticDiscriminantAnalysis	Bin-BOW	0.080	0.254	0.920	0.065	0.128
QuadraticDiscriminantAnalysis	Combined	0.079	0.235	0.760	0.065	0.268
QuadraticDiscriminantAnalysis	TF	0.074	0.212	0.710	0.064	0.338
QuadraticDiscriminantAnalysis	TF-IDF	0.067	0.186	0.623	0.055	0.361
DummyClassifier	TF	0.061	0.059	0.058	0.062	0.880
DummyClassifier	TF-IDF	0.061	0.059	0.058	0.062	0.880
DummyClassifier	Bin-BOW	0.061	0.059	0.058	0.062	0.880
DummyClassifier	BOW	0.061	0.059	0.058	0.062	0.880
DummyClassifier	LIWC	0.061	0.059	0.058	0.062	0.880
LabelPropagation	LIWC	0.024	0.006	0.005	0.600	0.934
LabelSpreading	Bin-BOW	0.024	0.006	0.005	0.600	0.934
LabelPropagation	Bin-BOW	0.024	0.006	0.005	0.600	0.934
LabelSpreading	Combined	0.024	0.006	0.005	0.600	0.934
LabelPropagation	Combined	0.024	0.006	0.005	0.600	0.934
LabelSpreading	BOW	0.024	0.006	0.005	0.600	0.934
LabelPropagation	BOW	0.024	0.006	0.005	0.600	0.934
LabelSpreading	LIWC	0.024	0.006	0.005	0.600	0.934
DummyClassifier	Combined	0.000	0.000	0.000	0.000	0.934
MultinomialNB	TF	0.000	0.000	0.000	0.000	0.934
MultinomialNB	TF-IDF	0.000	0.000	0.000	0.000	0.934

Table 2: Full table of VFP validation results

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
MLPClassifier	Bin-BOW	0.957	0.956	0.956	0.958	0.955
Base LR	BOW-BIN	0.956	0.956	0.956	0.956	0.955
LogisticRegression	Bin-BOW	0.956	0.956	0.956	0.956	0.955
Base MLP	BOW-BIN	0.955	0.933	0.927	0.964	0.944
RidgeClassifierCV	TF-IDF	0.955	0.950	0.948	0.957	0.952
LogisticRegression	BOW	0.950	0.903	0.889	0.969	0.929
Tuned LR	BOW-BIN	0.950	0.954	0.956	0.949	0.952
RidgeClassifierCV	TF	0.950	0.965	0.971	0.945	0.956
RidgeClassifierCV	Bin-BOW	0.947	0.942	0.941	0.948	0.944
MLPClassifier	BOW	0.946	0.942	0.941	0.949	0.944
Tuned SGD	TF	0.946	0.953	0.957	0.944	0.948
Tuned Soft Voting Ensemble	Mixed	0.944	0.963	0.971	0.938	0.952
SGDClassifier	TF	0.944	0.919	0.912	0.953	0.933
PassiveAggressiveClassifier	TF	0.942	0.958	0.964	0.937	0.948
LinearSVC	TF-IDF	0.942	0.941	0.941	0.944	0.941
PassiveAggressiveClassifier	TF-IDF	0.942	0.958	0.963	0.937	0.948
SGDClassifier	TF-IDF	0.941	0.940	0.941	0.943	0.941
RidgeClassifier	Combined	0.941	0.941	0.942	0.942	0.941
RidgeClassifier	Bin-BOW	0.941	0.941	0.941	0.941	0.941
MLPClassifier	TF-IDF	0.941	0.952	0.956	0.937	0.944
Base Soft Voting Ensemble	Mixed	0.940	0.968	0.978	0.932	0.952
Tuned MLP	BOW-BIN	0.940	0.951	0.956	0.937	0.944
CalibratedClassifierCV	TF-IDF	0.938	0.928	0.926	0.943	0.933
MLPClassifier	TF	0.938	0.962	0.971	0.930	0.948
Perceptron	TF	0.937	0.939	0.941	0.938	0.937
LinearSVC	Bin-BOW	0.934	0.933	0.934	0.935	0.933
CalibratedClassifierCV	Bin-BOW	0.934	0.933	0.934	0.935	0.933
Base SGD	TF	0.934	0.945	0.949	0.931	0.937
Perceptron	BOW	0.934	0.934	0.934	0.934	0.933
SGDClassifier	BOW	0.934	0.934	0.934	0.934	0.933
LinearSVC	TF	0.934	0.933	0.934	0.936	0.933
NuSVC	Bin-BOW	0.934	0.950	0.956	0.929	0.941
SGDClassifier	Bin-BOW	0.932	0.961	0.971	0.924	0.944
RidgeClassifier	TF-IDF	0.931	0.938	0.941	0.930	0.933
CalibratedClassifierCV	TF	0.930	0.920	0.919	0.935	0.926
Perceptron	TF-IDF	0.930	0.922	0.920	0.933	0.926
ExtraTreesClassifier	TF-IDF	0.923	0.948	0.956	0.916	0.933

Table 2 continued from previous page

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
Perceptron	Bin-BOW	0.922	0.958	0.971	0.911	0.937
RidgeClassifier	TF	0.921	0.942	0.949	0.915	0.929
SVC	Bin-BOW	0.920	0.935	0.942	0.916	0.926
ExtraTreesClassifier	BOW	0.920	0.936	0.942	0.915	0.926
RandomForestClassifier	TF	0.920	0.936	0.941	0.915	0.926
RandomForestClassifier	Combined	0.920	0.936	0.941	0.915	0.926
ExtraTreesClassifier	TF	0.915	0.934	0.942	0.910	0.922
ExtraTreesClassifier	Combined	0.914	0.934	0.941	0.908	0.922
PassiveAggressiveClassifier	Bin-BOW	0.907	0.926	0.934	0.902	0.914
MultinomialNB	BOW	0.906	0.948	0.963	0.894	0.922
ExtraTreesClassifier	Bin-BOW	0.906	0.937	0.948	0.897	0.918
RandomForestClassifier	TF-IDF	0.906	0.921	0.926	0.901	0.911
SVC	TF-IDF	0.904	0.920	0.926	0.900	0.911
LogisticRegression	TF	0.904	0.920	0.926	0.900	0.911
RandomForestClassifier	Bin-BOW	0.903	0.925	0.934	0.897	0.911
LinearSVC	BOW	0.902	0.870	0.860	0.915	0.888
CalibratedClassifierCV	BOW	0.902	0.829	0.808	0.931	0.874
XGBClassifier	Bin-BOW	0.902	0.898	0.897	0.903	0.900
NuSVC	TF-IDF	0.902	0.914	0.919	0.898	0.907
MultinomialNB	Bin-BOW	0.901	0.936	0.949	0.890	0.914
SVC	TF	0.899	0.919	0.926	0.894	0.907
NuSVC	TF	0.899	0.919	0.926	0.894	0.907
LogisticRegression	TF-IDF	0.897	0.902	0.904	0.896	0.900
AdaBoostClassifier	Bin-BOW	0.894	0.874	0.868	0.902	0.885
RandomForestClassifier	BOW	0.892	0.900	0.904	0.889	0.896
RidgeClassifierCV	Combined	0.886	0.872	0.868	0.892	0.881
AdaBoostClassifier	TF	0.885	0.916	0.927	0.875	0.896
XGBClassifier	TF-IDF	0.883	0.899	0.905	0.879	0.888
XGBClassifier	Combined	0.883	0.899	0.905	0.879	0.888
XGBClassifier	TF	0.882	0.893	0.898	0.880	0.885
PassiveAggressiveClassifier	BOW	0.879	0.863	0.860	0.886	0.870
XGBClassifier	BOW	0.876	0.886	0.890	0.874	0.878
AdaBoostClassifier	BOW	0.873	0.885	0.890	0.870	0.877
NearestCentroid	TF-IDF	0.866	0.822	0.809	0.884	0.852
NearestCentroid	TF	0.866	0.832	0.824	0.879	0.855
AdaBoostClassifier	Combined	0.865	0.888	0.897	0.859	0.874
AdaBoostClassifier	TF-IDF	0.865	0.888	0.897	0.859	0.874
BernoulliNB	TF-IDF	0.857	0.778	0.758	0.892	0.829
BernoulliNB	TF	0.857	0.778	0.758	0.892	0.829
BernoulliNB	Bin-BOW	0.857	0.778	0.758	0.892	0.829
BernoulliNB	Combined	0.857	0.778	0.758	0.892	0.829
BernoulliNB	BOW	0.857	0.778	0.758	0.892	0.829
NuSVC	BOW	0.857	0.819	0.809	0.872	0.844
BaggingClassifier	TF-IDF	0.855	0.831	0.824	0.864	0.844
BaggingClassifier	BOW	0.855	0.797	0.780	0.876	0.833
BaggingClassifier	TF	0.850	0.835	0.832	0.857	0.844
SVC	BOW	0.849	0.777	0.758	0.880	0.826
BaggingClassifier	Bin-BOW	0.845	0.822	0.817	0.855	0.837
MultinomialNB	TF	0.843	0.935	0.971	0.817	0.874
NearestCentroid	Bin-BOW	0.843	0.781	0.765	0.869	0.822
MultinomialNB	TF-IDF	0.842	0.945	0.985	0.813	0.877
KNeighborsClassifier	BOW	0.841	0.871	0.882	0.834	0.851
BaggingClassifier	Combined	0.832	0.836	0.839	0.833	0.829
LabelSpreading	TF	0.825	0.914	0.949	0.799	0.851
KNeighborsClassifier	TF	0.822	0.928	0.971	0.792	0.855
LogisticRegression	Combined	0.821	0.786	0.779	0.840	0.814
GaussianNB	Bin-BOW	0.817	0.815	0.816	0.819	0.814
DecisionTreeClassifier	TF-IDF	0.816	0.837	0.846	0.811	0.818
GaussianNB	Combined	0.814	0.930	0.978	0.781	0.848
LabelSpreading	TF-IDF	0.810	0.924	0.971	0.778	0.844
DecisionTreeClassifier	BOW	0.809	0.814	0.816	0.808	0.807
KNeighborsClassifier	TF-IDF	0.807	0.933	0.985	0.772	0.844
RidgeClassifierCV	BOW	0.806	0.728	0.706	0.837	0.785
LinearDiscriminantAnalysis	Bin-BOW	0.801	0.812	0.816	0.798	0.803
DecisionTreeClassifier	Combined	0.798	0.832	0.846	0.789	0.803
RidgeClassifier	BOW	0.781	0.705	0.683	0.810	0.762
DecisionTreeClassifier	Bin-BOW	0.776	0.773	0.772	0.778	0.773
KNeighborsClassifier	Bin-BOW	0.765	0.735	0.728	0.777	0.755
DecisionTreeClassifier	TF	0.755	0.788	0.802	0.747	0.758
GaussianNB	TF	0.748	0.732	0.729	0.756	0.744
LinearDiscriminantAnalysis	TF	0.746	0.812	0.838	0.727	0.758
GaussianNB	BOW	0.743	0.725	0.722	0.753	0.740
LogisticRegression	LIWC	0.735	0.728	0.728	0.740	0.729
NearestCentroid	BOW	0.734	0.528	0.486	0.857	0.699
GaussianNB	TF-IDF	0.728	0.699	0.693	0.742	0.722
ExtraTreeClassifier	TF-IDF	0.724	0.737	0.743	0.721	0.725
LinearDiscriminantAnalysis	TF-IDF	0.720	0.787	0.816	0.701	0.729

Table 2 continued from previous page

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
LinearDiscriminantAnalysis	Combined	0.720	0.787	0.816	0.701	0.729
ExtraTreeClassifier	Combined	0.716	0.697	0.692	0.724	0.710
CalibratedClassifierCV	Combined	0.710	0.637	0.617	0.738	0.695
ExtraTreeClassifier	TF	0.703	0.694	0.692	0.707	0.699
MultinomialNB	LIWC	0.694	0.750	0.772	0.678	0.695
ExtraTreeClassifier	Bin-BOW	0.689	0.736	0.757	0.678	0.688
LinearDiscriminantAnalysis	LIWC	0.686	0.589	0.568	0.737	0.673
RidgeClassifier	LIWC	0.686	0.589	0.568	0.737	0.673
LinearSVC	LIWC	0.679	0.740	0.778	0.671	0.669
CalibratedClassifierCV	LIWC	0.676	0.630	0.618	0.695	0.669
NuSVC	Combined	0.674	0.638	0.633	0.694	0.662
RidgeClassifierCV	LIWC	0.672	0.562	0.538	0.732	0.662
ExtraTreeClassifier	BOW	0.661	0.687	0.698	0.654	0.662
NuSVC	LIWC	0.646	0.594	0.581	0.669	0.636
RandomForestClassifier	LIWC	0.634	0.632	0.632	0.635	0.628
XGBClassifier	LIWC	0.622	0.596	0.588	0.631	0.617
ExtraTreesClassifier	LIWC	0.620	0.612	0.610	0.624	0.613
AdaBoostClassifier	LIWC	0.614	0.609	0.612	0.622	0.610
KNeighborsClassifier	Combined	0.609	0.647	0.661	0.598	0.602
KNeighborsClassifier	LIWC	0.609	0.647	0.661	0.598	0.602
LinearDiscriminantAnalysis	BOW	0.605	0.624	0.638	0.612	0.609
MultinomialNB	Combined	0.588	0.847	0.993	0.534	0.558
MLPClassifier	Combined	0.574	0.586	0.603	0.580	0.661
BernoulliNB	LIWC	0.573	0.843	1.000	0.517	0.528
BaggingClassifier	LIWC	0.571	0.533	0.522	0.585	0.569
DecisionTreeClassifier	LIWC	0.566	0.582	0.588	0.562	0.561
DummyClassifier	LIWC	0.561	0.836	1.000	0.506	0.506
DummyClassifier	TF-IDF	0.561	0.836	1.000	0.506	0.506
DummyClassifier	Combined	0.561	0.836	1.000	0.506	0.506
DummyClassifier	BOW	0.561	0.836	1.000	0.506	0.506
DummyClassifier	Bin-BOW	0.561	0.836	1.000	0.506	0.506
DummyClassifier	TF	0.561	0.836	1.000	0.506	0.506
CategoricalNB	TF-IDF	0.556	0.833	1.000	0.500	0.500
CategoricalNB	TF	0.556	0.833	1.000	0.500	0.500
SGDClassifier	LIWC	0.542	0.657	0.719	0.538	0.602
QuadraticDiscriminantAnalysis	TF-IDF	0.535	0.529	0.531	0.542	0.532
LinearSVC	Combined	0.533	0.613	0.646	0.511	0.636
ExtraTreeClassifier	LIWC	0.519	0.515	0.516	0.523	0.513
QuadraticDiscriminantAnalysis	TF	0.508	0.461	0.448	0.527	0.517
QuadraticDiscriminantAnalysis	LIWC	0.507	0.586	0.650	0.509	0.506
MLPClassifier	LIWC	0.489	0.567	0.615	0.476	0.577
Perceptron	LIWC	0.488	0.471	0.489	0.538	0.561
NearestCentroid	Combined	0.488	0.371	0.347	0.561	0.532
NearestCentroid	LIWC	0.488	0.371	0.347	0.561	0.532
QuadraticDiscriminantAnalysis	Combined	0.464	0.431	0.429	0.525	0.517
GaussianNB	LIWC	0.441	0.302	0.281	0.617	0.532
QuadraticDiscriminantAnalysis	BOW	0.429	0.366	0.352	0.471	0.501
QuadraticDiscriminantAnalysis	Bin-BOW	0.422	0.372	0.373	0.500	0.476
SVC	Combined	0.419	0.293	0.273	0.525	0.498
SGDClassifier	Combined	0.417	0.505	0.556	0.434	0.580
SVC	LIWC	0.416	0.321	0.310	0.525	0.502
Perceptron	Combined	0.368	0.389	0.415	0.380	0.553
PassiveAggressiveClassifier	LIWC	0.243	0.257	0.274	0.281	0.390
PassiveAggressiveClassifier	Combined	0.239	0.257	0.274	0.261	0.383
LabelSpreading	Combined	0.000	0.000	0.000	0.000	0.494
LabelSpreading	Bin-BOW	0.000	0.000	0.000	0.000	0.494
LabelSpreading	BOW	0.000	0.000	0.000	0.000	0.494
LabelSpreading	LIWC	0.000	0.000	0.000	0.000	0.494

Table 3: Full table of SBB validation results

Model	Feature Type	f0.5	f2.0	Precision	Recall	Accuracy
LabelSpreading	SSB	0.971	0.894	0.999	0.871	0.970
ExtraTreesClassifier	SSB	0.970	0.894	0.999	0.871	0.969
RandomForestClassifier	SSB	0.969	0.894	0.996	0.871	0.969
KNeighborsClassifier	SSB	0.967	0.893	0.994	0.871	0.968
XGBClassifier	SSB	0.964	0.893	0.990	0.871	0.968
MLPClassifier	SSB	0.963	0.893	0.989	0.871	0.968
BaggingClassifier	SSB	0.960	0.892	0.985	0.871	0.967
ExtraTreeClassifier	SSB	0.960	0.892	0.985	0.871	0.967
DecisionTreeClassifier	SSB	0.956	0.891	0.980	0.871	0.966
LogisticRegression	SSB	0.954	0.888	0.978	0.868	0.964
SGDClassifier	SSB	0.953	0.885	0.978	0.865	0.964
SVC	SSB	0.952	0.867	0.984	0.842	0.960
LinearSVC	SSB	0.951	0.884	0.976	0.864	0.963

Table 3 continued from previous page

Model	Feature Type	f0.5	f2.0	Precision	Recall	Accuracy
RidgeClassifier	SSB	0.937	0.861	0.966	0.838	0.955
LinearDiscriminantAnalysis	SSB	0.937	0.863	0.964	0.841	0.955
RidgeClassifierCV	SSB	0.936	0.850	0.969	0.825	0.953
Perceptron	SSB	0.926	0.878	0.943	0.863	0.955
MultinomialNB	SSB	0.903	0.767	0.960	0.730	0.929
PassiveAggressiveClassifier	SSB	0.898	0.849	0.918	0.837	0.940
AdaBoostClassifier	SSB	0.894	0.805	0.929	0.779	0.934
BernoulliNB	SSB	0.836	0.792	0.852	0.778	0.916
GaussianNB	SSB	0.826	0.801	0.834	0.793	0.914
CalibratedClassifierCV	SSB	0.785	0.488	0.984	0.433	0.865
NearestCentroid	SSB	0.753	0.763	0.750	0.767	0.885
QuadraticDiscriminantAnalysis	SSB	0.424	0.746	0.370	1.000	0.599
DummyClassifier	SSB	0.000	0.000	0.000	0.000	0.765

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
MLP	TF-IDF	0.978	0.974	0.973	0.979	0.997
Tuned MLP	TF IDF	0.977	0.976	0.975	0.978	0.997
Soft Voting Ensemble	TF-IDF	0.977	0.977	0.978	0.977	0.997
Tuned Soft Voting Ensemble	TF-IDF	0.976	0.978	0.979	0.976	0.997
Tuned SVM	TF-IDF	0.976	0.978	0.979	0.976	0.997
CCCV	TF-IDF	0.975	0.975	0.975	0.975	0.997
SVM	TF-IDF	0.971	0.950	0.943	0.979	0.995

(a) Validation results - SCI - Hyperparameter optimisation

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
Base LR	BOW-BIN	0.956	0.956	0.956	0.956	0.955
Base MLP	BOW-BIN	0.955	0.933	0.927	0.964	0.944
Tuned LR	BOW-BIN	0.950	0.954	0.956	0.949	0.952
Tuned SGD	TF	0.946	0.953	0.957	0.944	0.948
Tuned Soft Voting Ensemble	Mixed	0.944	0.963	0.971	0.938	0.952
Base Soft Voting Ensemble	Mixed	0.940	0.968	0.978	0.932	0.952
Tuned MLP	BOW-BIN	0.940	0.951	0.956	0.937	0.944
Base SGD	TF	0.934	0.945	0.949	0.931	0.937

(b) Validation results - VFP - Hyperparameter optimisation

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
Tuned Soft Voting Ensemble	SSB	0.970	0.894	0.871	0.998	0.969
Soft Voting Ensemble	SSB	0.969	0.894	0.871	0.996	0.969
RF	SSB	0.968	0.894	0.871	0.996	0.969
Tuned KNN	SSB	0.968	0.894	0.871	0.996	0.969
Tuned RF	SSB	0.967	0.893	0.871	0.995	0.969
KNN	SSB	0.967	0.893	0.871	0.994	0.968
Tuned XGB	SSB	0.966	0.893	0.871	0.992	0.968
XGB	SSB	0.964	0.893	0.871	0.990	0.968

(c) Validation results - SBB - Hyperparameter optimisation

Table 4: Validation results from hyperparameter optimisation

2 Test Results

There is an important note that needs to be made for the SCI test results. As SCI is a "constructed" step in the process of predicting predators and was not needed to be reported during the PAN-12 competition, it has resulted in some inconsistency in how people present these results. This is partially the reason for having it in the appendix since it is not a core part of the Thesis but important for others to check the work.

Not everyone adjusts their score for the predators removed by the prefilter. For the VFP module, this is required to make the results comparable to other researchers; however, since SCI is just a constructed task within another task, it is not as strict. However, to make the results presented here convertible to a format where others can compare it to their work, the data from the prefiltering is appended here.

The filter used removed conversations with less than 20 messages and conversations with only a single participant. The resulting number of conversations are as stated below.

Unfiltered Test 155128
Filtered Test 28559
Predatory Conversations in Unfiltered Test 3737
Predatory Conversations in Filtered Test 1474

Table 5: Full table of SCI test results

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
SVC	Bin-BOW	0.989	0.961	0.953	0.999	0.998
Tuned Soft Voting Ensemble	TF-IDF	0.987	0.961	0.953	0.997	0.997
Soft Voting Ensemble	TF-IDF	0.987	0.961	0.953	0.996	0.997
SGDClassifier	TF-IDF	0.987	0.965	0.958	0.994	0.998
SVC	TF	0.986	0.953	0.943	0.997	0.997
Tuned SVM	TF-IDF	0.985	0.962	0.954	0.994	0.997
SVC	TF-IDF	0.985	0.947	0.935	0.999	0.997
SGDClassifier	TF	0.985	0.953	0.942	0.996	0.997
MLPClassifier	TF-IDF	0.985	0.948	0.937	0.997	0.997
CalibratedClassifierCV	TF-IDF	0.984	0.959	0.951	0.993	0.997
LinearSVC	TF-IDF	0.984	0.954	0.944	0.994	0.997
CalibratedClassifierCV	TF	0.984	0.958	0.949	0.993	0.997
Tuned MLP	TF-IDF	0.984	0.950	0.939	0.996	0.997
LinearSVC	TF	0.984	0.958	0.949	0.993	0.997
LogisticRegression	Bin-BOW	0.984	0.948	0.936	0.996	0.997
MLPClassifier	Bin-BOW	0.984	0.955	0.946	0.994	0.997
MLPClassifier	TF	0.983	0.954	0.944	0.994	0.997
MLPClassifier	BOW	0.983	0.953	0.944	0.994	0.997
LinearSVC	Bin-BOW	0.982	0.954	0.944	0.992	0.997
CalibratedClassifierCV	Bin-BOW	0.982	0.945	0.933	0.994	0.996
PassiveAggressiveClassifier	TF-IDF	0.981	0.952	0.943	0.991	0.997
RidgeClassifier	TF-IDF	0.979	0.922	0.904	0.999	0.995
RidgeClassifierCV	TF-IDF	0.979	0.922	0.904	0.999	0.995
XGBClassifier	Bin-BOW	0.975	0.939	0.928	0.988	0.996
RidgeClassifier	TF	0.974	0.912	0.893	0.997	0.994
MLPClassifier	Combined	0.974	0.943	0.933	0.984	0.996
RidgeClassifierCV	TF	0.973	0.928	0.914	0.989	0.995
XGBClassifier	Combined	0.973	0.940	0.929	0.984	0.996
XGBClassifier	TF-IDF	0.972	0.929	0.916	0.988	0.995
RidgeClassifier	Combined	0.972	0.919	0.903	0.990	0.995
XGBClassifier	TF	0.971	0.921	0.905	0.989	0.995
SGDClassifier	Bin-BOW	0.971	0.963	0.960	0.973	0.997
PassiveAggressiveClassifier	TF	0.970	0.932	0.921	0.983	0.995
XGBClassifier	BOW	0.969	0.936	0.926	0.981	0.995
Perceptron	TF-IDF	0.966	0.935	0.925	0.976	0.995
PassiveAggressiveClassifier	Bin-BOW	0.965	0.934	0.924	0.976	0.995
RidgeClassifierCV	Bin-BOW	0.963	0.879	0.854	0.995	0.992
LogisticRegression	TF-IDF	0.961	0.864	0.837	0.998	0.992
LogisticRegression	BOW	0.960	0.930	0.920	0.971	0.994

Table 5 continued from previous page

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
LogisticRegression	TF	0.960	0.875	0.849	0.992	0.992
Perceptron	Bin-BOW	0.958	0.943	0.938	0.963	0.995
Perceptron	TF	0.953	0.930	0.923	0.961	0.994
RidgeClassifierCV	Combined	0.951	0.845	0.815	0.992	0.990
LinearSVC	BOW	0.947	0.934	0.930	0.951	0.994
AdaBoostClassifier	TF-IDF	0.942	0.915	0.906	0.952	0.993
AdaBoostClassifier	Combined	0.940	0.923	0.917	0.946	0.993
AdaBoostClassifier	TF	0.940	0.918	0.910	0.948	0.993
AdaBoostClassifier	BOW	0.935	0.903	0.893	0.947	0.992
SGDClassifier	BOW	0.934	0.929	0.927	0.936	0.993
MultinomialNB	Bin-BOW	0.934	0.873	0.854	0.957	0.991
AdaBoostClassifier	Bin-BOW	0.929	0.907	0.900	0.936	0.992
PassiveAggressiveClassifier	BOW	0.926	0.901	0.893	0.934	0.991
RidgeClassifier	Bin-BOW	0.916	0.865	0.849	0.934	0.989
BaggingClassifier	Combined	0.908	0.829	0.805	0.938	0.987
Perceptron	BOW	0.907	0.916	0.919	0.904	0.991
CalibratedClassifierCV	BOW	0.905	0.725	0.680	0.986	0.983
CalibratedClassifierCV	Combined	0.903	0.737	0.694	0.976	0.983
BaggingClassifier	TF-IDF	0.897	0.777	0.744	0.947	0.985
LinearSVC	Combined	0.886	0.908	0.915	0.879	0.989
BaggingClassifier	TF	0.881	0.748	0.712	0.936	0.983
BaggingClassifier	BOW	0.874	0.790	0.765	0.906	0.984
XGBClassifier	LIWC	0.869	0.766	0.737	0.910	0.983
BaggingClassifier	Bin-BOW	0.847	0.794	0.778	0.866	0.982
RidgeClassifierCV	BOW	0.843	0.734	0.704	0.886	0.980
KNeighborsClassifier	BOW	0.840	0.667	0.624	0.919	0.978
CalibratedClassifierCV	LIWC	0.836	0.626	0.577	0.941	0.976
LogisticRegression	Combined	0.825	0.745	0.722	0.855	0.979
ExtraTreesClassifier	Bin-BOW	0.824	0.539	0.483	1.000	0.973
MLPClassifier	LIWC	0.818	0.826	0.828	0.816	0.982
RandomForestClassifier	LIWC	0.815	0.584	0.533	0.939	0.974
BernoulliNB	Combined	0.803	0.523	0.468	0.979	0.972
LogisticRegression	LIWC	0.796	0.749	0.735	0.812	0.978
DecisionTreeClassifier	Combined	0.789	0.818	0.828	0.781	0.979
ExtraTreesClassifier	LIWC	0.785	0.526	0.474	0.940	0.971
RandomForestClassifier	Combined	0.785	0.477	0.422	1.000	0.970
BernoulliNB	BOW	0.783	0.482	0.427	0.989	0.970
BernoulliNB	TF	0.783	0.482	0.427	0.989	0.970
BernoulliNB	Bin-BOW	0.783	0.482	0.427	0.989	0.970
BernoulliNB	TF-IDF	0.783	0.482	0.427	0.989	0.970
AdaBoostClassifier	LIWC	0.782	0.728	0.712	0.801	0.976
BaggingClassifier	LIWC	0.780	0.615	0.575	0.856	0.973
ExtraTreesClassifier	BOW	0.776	0.464	0.409	1.000	0.970
RandomForestClassifier	Bin-BOW	0.769	0.455	0.400	1.000	0.969
DecisionTreeClassifier	TF-IDF	0.769	0.750	0.744	0.776	0.976
RandomForestClassifier	BOW	0.766	0.450	0.396	1.000	0.969
DecisionTreeClassifier	BOW	0.762	0.749	0.744	0.767	0.975
DecisionTreeClassifier	Bin-BOW	0.759	0.753	0.752	0.761	0.975
SVC	BOW	0.754	0.440	0.387	0.990	0.968
DecisionTreeClassifier	TF	0.751	0.757	0.759	0.749	0.974
PassiveAggressiveClassifier	Combined	0.750	0.625	0.592	0.803	0.972
KNeighborsClassifier	Combined	0.740	0.657	0.634	0.772	0.971
KNeighborsClassifier	LIWC	0.739	0.657	0.634	0.771	0.971
NearestCentroid	TF-IDF	0.711	0.882	0.959	0.668	0.973
ExtraTreesClassifier	Combined	0.699	0.367	0.317	1.000	0.965
SGDClassifier	LIWC	0.682	0.709	0.718	0.674	0.968
SGDClassifier	Combined	0.673	0.712	0.727	0.661	0.967
SVC	LIWC	0.665	0.369	0.321	0.910	0.963
SVC	Combined	0.661	0.364	0.317	0.909	0.963
LinearSVC	LIWC	0.660	0.347	0.300	0.942	0.963
KNeighborsClassifier	TF-IDF	0.656	0.853	0.948	0.609	0.966
RandomForestClassifier	TF-IDF	0.655	0.322	0.275	1.000	0.963
ExtraTreesClassifier	TF-IDF	0.651	0.318	0.271	1.000	0.962
RandomForestClassifier	TF	0.650	0.317	0.271	1.000	0.962
RidgeClassifier	BOW	0.638	0.691	0.710	0.622	0.963
MultinomialNB	BOW	0.635	0.872	0.996	0.582	0.963
ExtraTreesClassifier	TF	0.621	0.291	0.247	1.000	0.961
KNeighborsClassifier	Bin-BOW	0.606	0.444	0.408	0.690	0.960
Perceptron	LIWC	0.603	0.745	0.808	0.567	0.958
KNeighborsClassifier	TF	0.581	0.817	0.944	0.530	0.954
NearestCentroid	Bin-BOW	0.572	0.768	0.866	0.527	0.953
Perceptron	Combined	0.568	0.744	0.829	0.526	0.953
RidgeClassifier	LIWC	0.567	0.260	0.220	0.936	0.959
RidgeClassifierCV	LIWC	0.559	0.253	0.214	0.935	0.959
DecisionTreeClassifier	LIWC	0.548	0.576	0.586	0.539	0.953
NearestCentroid	TF	0.470	0.764	0.965	0.417	0.928
ExtraTreeClassifier	TF	0.458	0.440	0.434	0.465	0.945

Table 5 continued from previous page

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
ExtraTreeClassifier	TF-IDF	0.457	0.447	0.444	0.460	0.944
ExtraTreeClassifier	Bin-BOW	0.454	0.487	0.499	0.444	0.942
NearestCentroid	BOW	0.448	0.684	0.830	0.401	0.927
ExtraTreeClassifier	Combined	0.431	0.449	0.455	0.426	0.940
PassiveAggressiveClassifier	LIWC	0.422	0.698	0.893	0.373	0.917
ExtraTreeClassifier	BOW	0.422	0.444	0.453	0.415	0.939
ExtraTreeClassifier	LIWC	0.410	0.458	0.476	0.397	0.936
BernoulliNB	LIWC	0.159	0.389	0.752	0.133	0.733
MultinomialNB	Combined	0.153	0.339	0.571	0.129	0.780
MultinomialNB	LIWC	0.128	0.368	0.979	0.105	0.570
NearestCentroid	Combined	0.126	0.294	0.527	0.106	0.747
NearestCentroid	LIWC	0.126	0.294	0.527	0.106	0.747
DummyClassifier	TF	0.058	0.067	0.071	0.055	0.889
DummyClassifier	Bin-BOW	0.058	0.067	0.071	0.055	0.889
DummyClassifier	BOW	0.058	0.067	0.071	0.055	0.889
DummyClassifier	TF-IDF	0.058	0.067	0.071	0.055	0.889
MultinomialNB	TF	0.000	0.000	0.000	0.000	0.948
MultinomialNB	TF-IDF	0.000	0.000	0.000	0.000	0.948
DummyClassifier	Combined	0.000	0.000	0.000	0.000	0.948
DummyClassifier	LIWC	0.000	0.000	0.000	0.000	0.948

Table 6: Full table of VFP test results

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
RidgeClassifierCV	TF-IDF	0.947	0.848	0.819	0.986	1.000
SGDClassifier	TF-IDF	0.946	0.844	0.815	0.986	1.000
MLPClassifier	TF-IDF	0.946	0.844	0.815	0.986	1.000
MLPClassifier	TF	0.946	0.844	0.815	0.986	1.000
PassiveAggressiveClassifier	TF-IDF	0.945	0.860	0.835	0.977	1.000
Tuned Soft Voting Ensemble	Multi-Feature	0.944	0.828	0.795	0.990	1.000
Base Soft Voting Ensemble	Multi-Feature	0.944	0.837	0.807	0.986	1.000
LinearSVC	TF-IDF	0.944	0.847	0.819	0.981	1.000
MLPClassifier	BOW	0.941	0.818	0.783	0.990	1.000
MLPClassifier	Bin-BOW	0.938	0.811	0.776	0.990	1.000
SGDClassifier	TF	0.938	0.830	0.799	0.981	1.000
CalibratedClassifierCV	TF-IDF	0.937	0.827	0.795	0.981	1.000
Tuned SGD	TF	0.936	0.823	0.791	0.981	1.000
Base MLP	Bin-BOW	0.936	0.804	0.768	0.990	1.000
LinearSVC	TF	0.936	0.823	0.791	0.981	1.000
RidgeClassifierCV	TF	0.936	0.833	0.803	0.976	1.000
RidgeClassifier	Combined	0.935	0.839	0.811	0.972	1.000
RidgeClassifier	TF-IDF	0.935	0.839	0.811	0.972	1.000
SGDClassifier	Bin-BOW	0.933	0.835	0.807	0.972	1.000
PassiveAggressiveClassifier	TF	0.933	0.835	0.807	0.972	1.000
Perceptron	TF-IDF	0.932	0.842	0.815	0.967	1.000
Perceptron	Bin-BOW	0.931	0.819	0.787	0.976	1.000
CalibratedClassifierCV	TF	0.931	0.809	0.776	0.980	1.000
LogisticRegression	Bin-BOW	0.931	0.809	0.776	0.980	1.000
Base LR	Bin-BOW	0.931	0.800	0.764	0.985	1.000
Tuned LR	Bin-BOW	0.931	0.800	0.764	0.985	1.000
RidgeClassifier	TF	0.930	0.825	0.795	0.971	1.000
NuSVC	Bin-BOW	0.930	0.806	0.772	0.980	1.000
Tuned MLP	Bin-BOW	0.929	0.802	0.768	0.980	1.000
MultinomialNB	Bin-BOW	0.928	0.847	0.823	0.959	1.000
MLPClassifier	Combined	0.928	0.828	0.799	0.967	1.000
MultinomialNB	BOW	0.927	0.844	0.819	0.959	1.000
SVC	Bin-BOW	0.926	0.805	0.772	0.975	1.000
RidgeClassifier	Bin-BOW	0.926	0.795	0.760	0.980	1.000
Perceptron	TF	0.926	0.795	0.760	0.980	1.000
RidgeClassifierCV	Bin-BOW	0.924	0.798	0.764	0.975	1.000
ExtraTreesClassifier	TF	0.924	0.798	0.764	0.975	1.000
Base SGD	TF	0.923	0.814	0.783	0.966	1.000
SVC	TF-IDF	0.922	0.811	0.779	0.966	1.000
CalibratedClassifierCV	Bin-BOW	0.920	0.798	0.764	0.970	1.000
AdaBoostClassifier	Bin-BOW	0.920	0.814	0.783	0.961	1.000
LogisticRegression	BOW	0.919	0.768	0.728	0.984	1.000
LinearSVC	Bin-BOW	0.919	0.794	0.760	0.970	1.000
RandomForestClassifier	Combined	0.918	0.800	0.768	0.965	1.000
NuSVC	TF-IDF	0.918	0.800	0.768	0.965	1.000
ExtraTreesClassifier	Bin-BOW	0.917	0.797	0.764	0.965	1.000
RandomForestClassifier	Bin-BOW	0.915	0.800	0.768	0.961	1.000
ExtraTreesClassifier	TF-IDF	0.914	0.806	0.776	0.956	1.000
SGDClassifier	BOW	0.913	0.787	0.752	0.965	1.000
Perceptron	BOW	0.913	0.787	0.752	0.965	1.000
SVC	TF	0.910	0.796	0.764	0.956	1.000

Table 6 continued from previous page

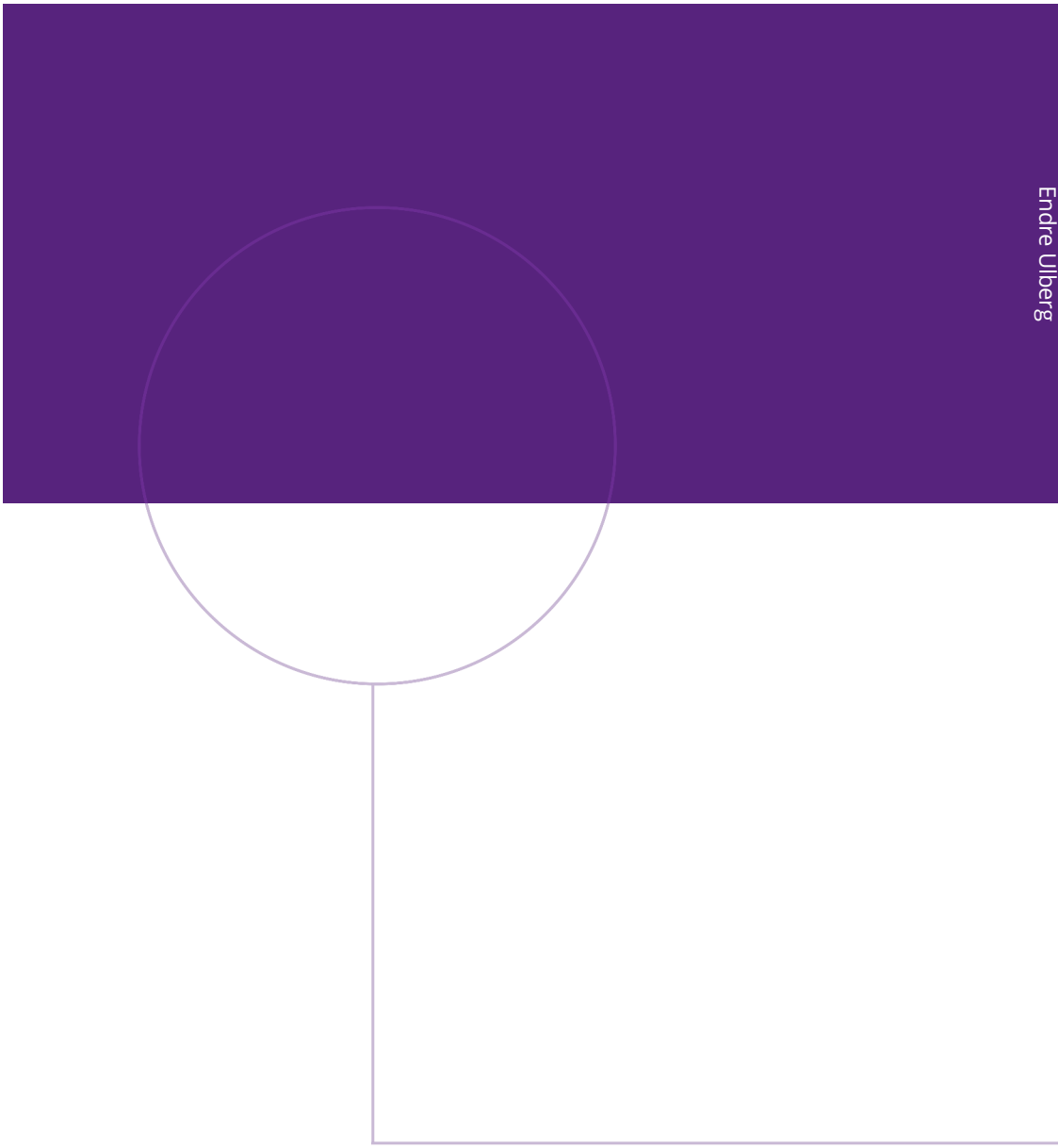
Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
RandomForestClassifier	BOW	0.909	0.776	0.740	0.964	1.000
CalibratedClassifierCV	BOW	0.909	0.750	0.709	0.978	1.000
XGBClassifier	BOW	0.909	0.792	0.760	0.955	1.000
LogisticRegression	TF-IDF	0.908	0.773	0.736	0.964	1.000
AdaBoostClassifier	TF	0.906	0.795	0.764	0.951	1.000
XGBClassifier	Bin-BOW	0.906	0.795	0.764	0.951	1.000
ExtraTreesClassifier	Combined	0.904	0.798	0.768	0.947	1.000
RandomForestClassifier	TF-IDF	0.904	0.798	0.768	0.947	1.000
ExtraTreesClassifier	BOW	0.903	0.794	0.764	0.946	1.000
LinearSVC	BOW	0.902	0.765	0.728	0.959	1.000
RandomForestClassifier	TF	0.900	0.778	0.744	0.950	1.000
NuSVC	TF	0.899	0.784	0.752	0.946	1.000
LogisticRegression	TF	0.899	0.784	0.752	0.946	1.000
AdaBoostClassifier	BOW	0.898	0.781	0.748	0.945	1.000
XGBClassifier	TF	0.898	0.797	0.768	0.938	1.000
LogisticRegression	Combined	0.895	0.757	0.721	0.953	1.000
XGBClassifier	TF-IDF	0.895	0.790	0.760	0.937	1.000
XGBClassifier	Combined	0.895	0.790	0.760	0.937	1.000
PassiveAggressiveClassifier	BOW	0.892	0.799	0.772	0.929	1.000
PassiveAggressiveClassifier	Bin-BOW	0.884	0.784	0.756	0.923	1.000
AdaBoostClassifier	Combined	0.881	0.761	0.728	0.930	1.000
AdaBoostClassifier	TF-IDF	0.881	0.761	0.728	0.930	1.000
MultinomialNB	TF-IDF	0.880	0.855	0.847	0.888	1.000
NearestCentroid	TF	0.879	0.742	0.705	0.937	1.000
MultinomialNB	TF	0.878	0.851	0.843	0.888	1.000
BernoulliNB	TF	0.878	0.685	0.638	0.970	1.000
BernoulliNB	TF-IDF	0.878	0.685	0.638	0.970	1.000
BernoulliNB	BOW	0.878	0.685	0.638	0.970	1.000
BernoulliNB	Combined	0.878	0.685	0.638	0.970	1.000
BernoulliNB	Bin-BOW	0.878	0.685	0.638	0.970	1.000
NearestCentroid	TF-IDF	0.873	0.711	0.669	0.944	1.000
BaggingClassifier	TF-IDF	0.872	0.740	0.705	0.927	1.000
BaggingClassifier	Combined	0.871	0.737	0.701	0.927	1.000
BaggingClassifier	TF	0.865	0.746	0.713	0.914	1.000
DecisionTreeClassifier	TF	0.862	0.745	0.713	0.909	1.000
KNeighborsClassifier	BOW	0.851	0.733	0.701	0.899	1.000
NearestCentroid	Bin-BOW	0.849	0.673	0.630	0.930	1.000
RidgeClassifierCV	Combined	0.847	0.770	0.748	0.876	1.000
DecisionTreeClassifier	Combined	0.846	0.723	0.689	0.897	1.000
BaggingClassifier	Bin-BOW	0.845	0.686	0.646	0.916	1.000
KNeighborsClassifier	TF	0.845	0.831	0.827	0.850	1.000
NuSVC	BOW	0.845	0.693	0.653	0.912	1.000
BaggingClassifier	BOW	0.840	0.669	0.626	0.919	1.000
KNeighborsClassifier	TF-IDF	0.840	0.836	0.835	0.841	1.000
DecisionTreeClassifier	TF-IDF	0.830	0.700	0.665	0.885	1.000
SVC	BOW	0.822	0.639	0.595	0.910	1.000
DecisionTreeClassifier	BOW	0.802	0.708	0.681	0.840	1.000
DecisionTreeClassifier	Bin-BOW	0.788	0.673	0.642	0.836	0.999
MultinomialNB	LIWC	0.756	0.676	0.653	0.787	0.999
LogisticRegression	LIWC	0.754	0.634	0.602	0.805	0.999
ExtraTreeClassifier	TF-IDF	0.753	0.637	0.606	0.802	0.999
KNeighborsClassifier	Bin-BOW	0.752	0.563	0.520	0.846	0.999
RidgeClassifierCV	BOW	0.736	0.522	0.476	0.852	0.999
ExtraTreeClassifier	BOW	0.723	0.593	0.559	0.780	0.999
ExtraTreeClassifier	Bin-BOW	0.714	0.604	0.575	0.760	0.999
ExtraTreeClassifier	TF	0.714	0.554	0.516	0.789	0.999
RandomForestClassifier	LIWC	0.702	0.596	0.567	0.746	0.999
RidgeClassifier	BOW	0.693	0.492	0.449	0.803	0.999
ExtraTreeClassifier	Combined	0.693	0.616	0.595	0.723	0.999
CalibratedClassifierCV	LIWC	0.691	0.470	0.425	0.818	0.999
RidgeClassifier	LIWC	0.690	0.492	0.449	0.797	0.999
XGBClassifier	LIWC	0.683	0.573	0.543	0.730	0.999
RidgeClassifierCV	LIWC	0.682	0.477	0.433	0.797	0.999
AdaBoostClassifier	LIWC	0.680	0.595	0.571	0.714	0.999
LinearSVC	Combined	0.678	0.647	0.638	0.689	0.999
NearestCentroid	BOW	0.677	0.421	0.374	0.848	0.999
BaggingClassifier	LIWC	0.676	0.562	0.531	0.726	0.999
MLPClassifier	LIWC	0.674	0.490	0.449	0.770	0.999
CalibratedClassifierCV	Combined	0.672	0.450	0.406	0.805	0.999
LinearSVC	LIWC	0.669	0.627	0.614	0.684	0.999
ExtraTreesClassifier	LIWC	0.662	0.569	0.543	0.701	0.999
MultinomialNB	Combined	0.658	0.795	0.854	0.622	0.999
ExtraTreeClassifier	LIWC	0.628	0.556	0.535	0.657	0.999
KNeighborsClassifier	Combined	0.619	0.535	0.512	0.653	0.999
KNeighborsClassifier	LIWC	0.619	0.535	0.512	0.653	0.999
DecisionTreeClassifier	LIWC	0.617	0.538	0.516	0.648	0.999
DummyClassifier	Combined	0.610	0.779	0.858	0.569	0.999

Table 6 continued from previous page

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
SGDClassifier	Combined	0.610	0.779	0.858	0.569	0.999
DummyClassifier	TF	0.610	0.779	0.858	0.569	0.999
DummyClassifier	TF-IDF	0.610	0.779	0.858	0.569	0.999
DummyClassifier	LIWC	0.610	0.779	0.858	0.569	0.999
BernoulliNB	LIWC	0.610	0.779	0.858	0.569	0.999
SGDClassifier	LIWC	0.610	0.779	0.858	0.569	0.999
DummyClassifier	Bin-BOW	0.610	0.779	0.858	0.569	0.999
DummyClassifier	BOW	0.610	0.779	0.858	0.569	0.999
NuSVC	LIWC	0.607	0.339	0.295	0.824	0.999
NuSVC	Combined	0.607	0.335	0.291	0.832	0.999
PassiveAggressiveClassifier	LIWC	0.592	0.730	0.791	0.557	0.999
SVC	Combined	0.460	0.249	0.216	0.639	0.999
NearestCentroid	LIWC	0.454	0.295	0.264	0.554	0.999
NearestCentroid	Combined	0.454	0.295	0.264	0.554	0.999
SVC	LIWC	0.403	0.198	0.169	0.614	0.999
PassiveAggressiveClassifier	Combined	0.132	0.052	0.043	0.268	0.999
Perceptron	LIWC	0.122	0.034	0.028	0.875	0.999
Perceptron	Combined	0.122	0.034	0.028	0.875	0.999

Table 7: Full table of SBB test result

Model	Feature Type	f0.5	f2.0	Recall	Precision	Accuracy
RandomForestClassifier	SBB	0.952	0.833	0.799	1.000	1.000
DecisionTreeClassifier	SBB	0.952	0.833	0.799	1.000	1.000
LabelSpreading	SBB	0.952	0.833	0.799	1.000	1.000
ExtraTreesClassifier	SBB	0.952	0.833	0.799	1.000	1.000
MLPClassifier	SBB	0.952	0.833	0.799	1.000	1.000
XGBClassifier	SBB	0.935	0.829	0.799	0.976	1.000
Tuned Soft Voting Ensemble	SBB	0.935	0.829	0.799	0.976	1.000
Soft Voting Ensemble	SBB	0.928	0.828	0.799	0.967	1.000
BaggingClassifier	SBB	0.921	0.827	0.799	0.958	1.000
Tuned XGB	SBB	0.911	0.825	0.799	0.944	1.000
Tuned RF	SBB	0.911	0.825	0.799	0.944	1.000
KNeighborsClassifier	SBB	0.839	0.809	0.799	0.849	1.000
Tuned KNN	SBB	0.833	0.807	0.799	0.842	1.000
CalibratedClassifierCV	SBB	0.709	0.424	0.374	0.913	0.999
LinearSVC	SBB	0.666	0.750	0.783	0.642	0.999
LogisticRegression	SBB	0.629	0.746	0.795	0.598	0.999
SVC	SBB	0.624	0.731	0.776	0.595	0.999
SGDClassifier	SBB	0.572	0.725	0.795	0.534	0.999
RidgeClassifierCV	SBB	0.472	0.659	0.760	0.431	0.999
RidgeClassifier	SBB	0.452	0.656	0.772	0.410	0.998
LinearDiscriminantAnalysis	SBB	0.433	0.646	0.772	0.390	0.998
MultinomialNB	SBB	0.376	0.560	0.669	0.339	0.998
Perceptron	SBB	0.255	0.522	0.799	0.218	0.996
AdaBoostClassifier	SBB	0.241	0.476	0.705	0.207	0.997
PassiveAggressiveClassifier	SBB	0.165	0.406	0.791	0.138	0.994
BernoulliNB	SBB	0.129	0.334	0.713	0.107	0.993
GaussianNB	SBB	0.119	0.319	0.728	0.098	0.992
NearestCentroid	SBB	0.072	0.221	0.705	0.059	0.987
QuadraticDiscriminantAnalysis	SBB	0.015	0.057	0.917	0.012	0.913
DummyClassifier	SBB	0.000	0.000	0.000	0.000	0.999



Endre Ulberg

NTNU
Norwegian University of
Science and Technology