Classifying Legally Actionable Threats Using Language Models

Universiteit Utrecht

Noud Jan de Rijk (5670721)

Faculty of Science

Utrecht University

Supervisor

Dr. Pablo Mosteiro Romero & Dr. Dong Nguyen

In partial fulfillment of the requirements for the degree of

Master of Science in Business Informatics

March 23, 2022

Acknowledgements

Firstly, I would like to wholeheartedly thank my first supervisor, dr. Pablo Mosteiro Romero, for his guidance, patience, and sharpness. Even though the starting point for this thesis was quite eventful, you helped me by providing a clear eye on the situation and allowing me to widen my gaze. Additionally, each weekly meeting helped me further with completing my thesis and this was not possible without your help.

Secondly, I would like to thank my second supervisor, dr. Dong Nguyen, for her feedback and allowing me to get new perspectives on my work.

Thirdly, I would like to thank my daily supervisor at the National Police, Janneke Veltkamp, for her continued support in my thesis and helping me finding my way at the National Police.

Lastly, I would like to thank the National Police of the Netherlands for allowing me to have an internship there and allowing me to write my thesis collaboratively.

Dedication

This thesis is dedicated to my mother, Petra de Rijk-Kruiderink, a smart and caring woman whom I still miss at every waking moment.

Abstract

Threat classification is a relatively new research field in the Natural Language Processing domain. It pertains to models attempting to classify what texts constitute a threat and which texts do not. This is an essential research field as uttering threats is illegal as opposed to insulting someone.

This research operationalizes the Dutch legal definition of what constitutes a threat and investigates to what extent a language model can classify legally actionable threats from texts. Language models are the state-of-the-art technique for numerous NLP tasks, including text classification. In the text classification domain, it allows a Machine Learning (ML) model to be pre-trained on millions of tokens before fine-tuning the model on a downstream task. In this way, a language model is created that learns the syntax of a language. This pre-training negates the problem of data scarcity, which is a recurring problem in threat classification. In this study, the application of a language model is compared to previously used models in the threat classification domain (i.e. BiLSTM, CNN, Naive Bayes, & SVM). The performance metrics that the models are compared with are F1-scores and Precision-Recall Area-Under-Curve (PR-AUC) score. All models are trained on publicly available datasets containing threats and non-threats that are manually re-annotated. Additionally, the models are evaluated with two datasets, namely a Dutch dataset and an English dataset. The goal of these models is to predict whether a threat that is uttered is legally actionable. The models were evaluated

by means of a stratified 10-fold split.

The results of the study are that it is possible to operationalize the Dutch legal definition by means of annotation guidelines. Two annotators re-annotated a Dutch and English threat dataset and their agreement was not caused by chance and the Dutch dataset was deemed sufficient for the target institution (i.e. the National Police). The language model subsequently statistically significantly outperformed the benchmark models in the majority of performance metrics.

Contents

Chapter 1

Introduction

This study aims to investigate the application of language models for threat classification in, primarily, the online domain. The topic will be introduced through a problem description, followed by a case description, research questions, and an outline of the remaining research proposal is given.

1.1 Problem description

With the tremendous adoption of social media and subsequent online conversations, an increasing number of people experience toxic messages. According to Georgakopoulus et al., a toxic message is defined as "not only the verbal violence but a comment that is rude, disrespectful or otherwise likely to make someone leave a discussion" [1]. In addition, categories of toxic messages include spam, hate speech, insults, fake news, and threats.

In recent years, research into the classification of toxic comments on social networks has become an active research field [2; 3; 4; 5]. However, not much research has been done on a severe subcategory of toxic comments, namely threats [6; 7; 8]. Threats of intending to hurt one's body or property are illegal and legally

actionable as opposed to spam, insults, fake news, and otherwise disrespectful online discourse. Therefore, it is crucial to research the classification of legally actionable threats (LATs).

When examining existing studies in the domain of threat classification, it can be concluded that in the majority of studies the authors do not use a legally actionable definition for what constitutes an illegal threat [6; 7; 8; 9; 10]. Many authors do not even use a definition or leverage annotation guidelines to produce a consistent data set. For these reasons, the results of these studies are impossible to generalize across the domain. To facilitate generalization of results within this domain in the future, a consistent definition of an illegal threat must be operationalized. This study operationalizes the legal definition of a threat and applies it algorithmically. In doing so, the first step is made to generalize results across the domain in the future.

Recent studies in the NLP (Natural Language Processing) domain have seen a focus shift from simple classifiers and recurrent & convolutional neural network models to models that are pre-trained on one or multiple languages [11]. The reason is that these pre-trained language models address a gap in knowledge by transferring knowledge of the languages that they are pre-trained on. This has been proven to negate data scarcity [12], which is a recurring problem in the threat classification domain [6; 9].

In this study threats on social media and threatening letters are classified based on the Dutch legal definition. This is done by constructing two corpora, one in English and one in Dutch, with manual span-wise annotations of alleged threats. Two corpora are constructed in different languages as the data that are publicly available for Dutch is homogeneous, meaning that it comes from a single source and is almost wholly a singular type of threat (i.e. death threat). The Dutch data are comprised of tweets containing trivial messages and death threats. Since this

discrepancy can impact the results negatively, an additional corpus is constructed in English. This corpus is constructed using multiple sources containing different types of threats as well as a range of toxic & non-toxic comments. These corpora are then manually annotated to indicate if a legally actionable threat exists within the text and if so, which words signify this threat. This, to my best knowledge, has not been investigated in prior studies.

Additionally, the efficacy of a state-of-the-art language model is compared to that of models that have been previously used in the threat classification domain [6; 7; 8; 9; 10]. These previously used models entail a Bi-directional Long Short-Term Memory (BiLSTM), Convolutional Neural Network (CNN), Naive Bayes (NB) model, and Support Vector Machine (SVM) model. These models are used as benchmark models to compare the language model with. This, to my best knowledge, has also not been investigated in prior studies.

1.2 Case description

This research was conducted in cooperation with the National Police (NP) of the Netherlands where I was active within the Data Intelligence Team (DIT). The reason for cooperating with the National Police is that they have a two-sided interest in this problem.

Firstly, the NP currently do not have the means to classify threats using a legal definition. Secondly, threats can be the precursor for a potentially dangerous situation. Therefore, it is of interest to the NP that there are means to survey online discourse for assessing potentially dangerous situations for people and the public order. Using a classification model, online discourse can be filtered to create a collection of messages that must be further investigated. By cooperating with the NP, I can validate my work on their internal data comprised of Dutch & English

threats. In doing so, it is possible to validate the performance and robustness of a language model on unseen data. Due to restrictions, I am not able to provide any examples of the data that I use to validate my work, but I can provide meta-data and performance metrics.

1.3 Research questions

The main research question (MRQ) of this thesis is as follows:

MRQ: To what extent can the Dutch legal definition of a threat be learned algorithmically by language models?

To find an answer to the MRQ, the following sub-questions (RQ_n) need to be answered:

RQ_1: Is the inter-rater reliability (IRR) in terms of Cohen's Kappa caused by chance agreement and is it sufficient for the annotation of LATs?

The goal of RQ_1 is to investigate if it is possible to annotate threats using the Dutch legal definition and to test if the IRR in terms of Cohen's Kappa is sufficient for use in the legal domain. Regarding sufficiency, the literature forms a starting point to interpret the Cohen's Kappa score and gauge its sufficiency [13; 14; 15; 16]. It is possible that the IRR is caused by chance agreement, this means that annotators do not agree consistently on the majority of cases. However, it is hypothesised that the Cohen's Kappa is not caused due to chance agreements. Furthermore, the potential disagreements between annotators must be investigated to check the operationalisation of the legal definition. Lastly, a scientific contribution is made when assessing whether a legal definition for a legally actionable threat is reproducible as this has not been done before in the scientific domain.

RQ_2: Do the language model and benchmark models differ statistically signif-

icantly in terms of their F1-score and PR-AUC score with respect to the classification of LATs?

The goal of RQ_2 is to investigate if a significant difference exists between the language model and the chosen benchmark models in terms of performance. In this case, performance is measured by means of F1-score and Precision-Recall Area Under Curve (PR-AUC) score. It is hypothesised that the language model significantly outperforms the benchmark models in terms of F1-score and PR-AUC score. The reason for this is that it has achieved state-of-the-art performance in numerous NLP tasks, including text classification [17; 18]. Additionally, language models are claimed to perform better at tasks where there is data scarcity [19; 20]. An important scientific contribution could be made by investigating the performance difference between these models in relation to threat classification. This approach has not yet been taken by the scientific community. The answer to this question could provide institutions with more insight into how to classify LATs.

RQ_3: Do the language model and benchmark models differ statistically significantly in terms of their F1-score and PR-AUC score with respect to the classification of LATs in internal police data?

The goal of RQ_3 is to investigate if a significant difference exists between the language model and the benchmark models in terms of performance when classifying LATs on unseen data. More specifically, internal police data is used. It is assumed that the unseen police data can be dissimilar to the data that the models have been trained on. In doing so, the models' performances on real-world data can be gauged. It is hypothesised that the language model performs significantly better than the benchmark models. An important scientific contribution could be made by investigating whether language models are a suitable improvement upon other existing methods to classify threat as this has not yet been pursued by the

scientific community.

RQ_4: When does the language model correctly and incorrectly classify threats and non-threats?

The goal of RQ_4 is to explain the language model's inner workings and behaviour when correctly and incorrectly classifying LATs. It is imperative to investigate how the language model comes to its conclusions. In doing so, its aptitude for determining what constitutes and does not constitute a threat following the legal definition is evaluated. Additionally, it is not unheard of that language models, that are pre-trained on terabytes of data, introduce societal biases that are present on the texts that they are trained on [21]. Therefore, it must be investigated whether the societal biases skew the results of this study.

1.4 Thesis proposal outline

The remainder of this thesis proposal is structured as follows: Chapter 2 reviews the existing literature related to this subject; and Chapter 4 describes the methodology for how the research will be conducted, the possible risks that the thesis faces & their possible mitigation, and a planning with milestones & deadlines for the rest of the thesis project; Chapter 3 elaborates upon the data set that will be used for this thesis; Chapter 5 details the results as acquired from the specified methodology; and finally Chapter 6 presents the conclusions of the research, highlights the limitations of the work, and outlines what future work should entail.

Chapter 2

Literature review

This chapter reviews the relevant literature for this thesis. First, the literature review protocol is discussed. Second, the literature review is given where a domain understanding is given alongside a review of current literature on threat classification.

2.1 Literature review protocol

This literature review summarises state-of-the-art text classification techniques and the current knowledge of online threat classification.

This literature review is set up in three sections. Section 2.2 provides an overview of the domain of legal threats and how they are assessed in the literature. Section 2.3 reviews the existing literature on text classification techniques. Section 2.4 presents the current knowledge of online threat classification.

To acquire the necessary literature to summarise and compare existing works in both sections, the following techniques were used:

1. Forward snowballing on influential papers[22]
2. Systematic literature search [23]

3. Expert recommendations

First, a forward snowballing technique is used to gain a deep understanding of the legal domain concerning threats, existing text classification techniques, and how online threats are currently being classified [22]. First, a primary set of papers that is deemed influential in its respective domain is identified. This primary set of papers is identified by looking at papers' titles, abstract, and amount they have been cited. To make sure that these papers are deemed influential in their respective domains, a minimum number of citations is set according to the breadth of the domain [22]. Therefore, a minimum of 10 citations was set as a criterion for the legal domain of threats, a minimum of 200 citations was set as the criterion for the text classification domain, and a minimum of 20 citations was set as a criterion for the online threat classification domain. Second, subsequent papers are identified by looking at papers referenced by this primary set of papers. By reading the found paper's abstract and full text, the paper is included only when it makes a relevant contribution to the objectives of this research. Using this forward snowballing technique, a total of 42 relevant papers were identified that form the basis of the literature review.

The starting set of influential papers was as follows:

- "Attention is all you need" by Vaswani et al. [24]
- "Deep Learning–based Text Classification: A Comprehensive Review" by Minaee et al. [25]
- "Threatening Stances: a corpus analysis of realized vs. non-realized threats" by Gales [26].
- "Resources and benchmark corpora for hate speech detection: a systematic review" by Poletto et al. [27].

Second, a systematic literature search was conducted in addition to the forward snowball method. This systematic literature search is used to find relevant literature for threat classification in the online domain that may have been overlooked during the forward snowballing search [23]. A list of keywords is constructed based on the context of this study and themes recognized from the forward snowballing search. This list can be observed in Appendix A. Based on these keywords, a set of queries are performed on several academic search engines. The search engines used were:

- ACM Digital Library
- Institute of Electrical and Electronics Engineers (IEEE)
- PubMed
- arXiv

The queries' results can be observed in Appendix A. To include only relevant papers from this systematic literature search, a set of inclusion criteria are used. This set of inclusion criteria can be observed in Appendix A. After filtering the queries' results with the inclusion criteria, a total of 38 relevant papers were left and added to the systematic literature review results.

Lastly, the final set of relevant papers for this study's literature review is enriched by expert recommendations. These experts are characterized as researchers in the fields of NLP, sentiment analysis, and/or criminal law. A total of 9 papers were added to the literature review using this method.

2.2 Domain understanding

This section presents a summary of the existing body of work surrounding threatening texts. This is done by giving an overview of works that present a workable

definition of what constitutes a threat in Section 2.2.1. Additionally, an overview is given of linguistic markers that threatening texts share in Section 2.2.2.

2.2.1 Definition of a threat

This section follows the seminal work by Fraser [28]. Fraser states that threats can be a variety of things; threats can come from anger (i.e. "I will cut you because you deceived me!") or humor (i.e. "Timmy, when I feed you apple seeds, you will grow a plant in your stomach that will burst your belly!"). Therefore, some threats are perfectly legal [28]. Types of threats that are typically illegal are, for example, threats of bribery, extortion, robbery, and assault (i.e. threatening to cause physical injury) [28]. In addition, Fraser provides three parameters that must be fulfilled for a threat to be made successfully (i.e. both for legal and illegal threats):

- C1. The speaker's intention to personally commit an act (or be responsible for bringing about the commission of the act);
- C2. The speaker's belief that this act will result in an unfavourable state of the world for the addressee.
- C3. The speaker's intention to intimidate the addressee through the addressee's awareness of the intention in C1.

Fraser refers to Black Law's Dictionary's definition of a threat that reflects the aforementioned parameters for the illegal types of threats: [28]

> Threat: a declaration of intention or determination to inflict punishment, loss, or pain on another, or to injure another by the commission of some unlawful act. U.S. v. Daulong, D.C.La., 60F.Supp. 235, 236. A menace; especially, any menace of such a nature and extent as to

> unsettle the mind of the person on whom it operates, and to take away from his acts that free and voluntary action which alone constitutes consent. (Abbott, United States v. French, D.C.Fla., 243 F. 785, 786 (1651))

In the Dutch penal code, a threat is defined in Article 285, section 1 (translated with DeepL [1]: [29]

> Threat of openly committing violence in association against persons or property, of violence against an internationally protected person or his protected property, of any crime causing danger to the general security of persons or property or common danger to the provision of services, of rape, of actual indecency assault, of any crime against life, of hostage-taking, of aggravated assault or of arson, shall be punishable by imprisonment for a term not exceeding two years or a fine of the fourth category.

The difference with the definition of an illegal threat as defined in Black Law's Dictionary [30], is that the Dutch penal code specifically mentions arson, danger to the public order, and indecent assault [31] as illegal acts to threaten with [29]. Furthermore, the Dutch penal code specifically mentions internationally protected persons [32], their property/goods, and public services as entities that could be under threat.

2.2.2 Linguistic markers of illegal threats

Fraser argues that threats are part of commissive illocutionary speech acts as defined by Searle [28; 33]. Commissive illocutionary acts are linguistic acts (i.e.

[1] https://www.deepl.com/translator

speaking or writing something) in which the utterer of the words can be said to do something. Therefore, threats are somewhat distinguishable from other commissives, such as denying something because threats are bound to intimidate the addressee and the utterer of the threat expresses the intention and not the commitment to perform the act [28].

Fraser presents the conclusion that although threats share common syntaxes, it is probably impossible to determine whether a text is threatening by the language (e.g. text) alone. This conclusion is supported in later years by other studies [34; 35].

In contrast, a study by Nini investigated the malicious forensic texts (MFT) corpus which is comprised of texts containing threats, abuse, and spread of malicious information [36]. MFTs were investigated to ascertain "which linguistic features are typical and pervasive of the situational characteristics of MFTs" [36]. To realize this, the following information was required:

- Description of the situational characteristics of texts (i.e. text types as defined by Biber [37])
- Description of the language (e.g. what types of threats are being made)
- Functional interpretation of the connection between linguistic features as defined by Biber [38] and situational context as defined by Biber

Based on this information for each MFT, Nini concludes that it is possible, in contrast to Fraser's conclusion [28], to distinguish threatening texts from non-threatening texts. The linguistic feature named Overt Expression of Persuasion, marks the author's point of view and their assessment of certainty that a given event will occur. For example, it was shown that threats that contain many infinitives (e.g. "I *want to* murder you"), suasive verbs (e.g. "*I demand* that you stop

talking or I will murder you") , and split auxiliaries (e.g. I will *slowly* torture you). This linguistic feature makes it possible to distinguish threats from non-threats on language alone. This finding is supported by Gales [39].

2.3 Overview of Text Classification techniques

NLP "is a theoretically motivated range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis to achieve human-like language processing for a range of tasks or applications" [40]. These NLP tasks range from question answering, speech recognition, inference, summarising texts, translation, text classification, and more [40]. This research will focus on text classification methods. Text classification is the process of assigning one or more labels to a text.

For this literature review, text classification techniques are divided into two categories: simple and complex. This distinction was made based on notions by Hacker et al. [41] regarding the performance/explainability trade-off in algorithmic models and parameters like simulatability, decomposability, and transparency as given by Arrieta et al. [42].

2.3.1 Simple text classification methods

For simple text classification methods, rule-based models, distance-based models, and tree-based models are considered. These methods are reviewed on a technical level.

2.3.1.1 Rule-based models

Rule-based models are based on the principles of there being numerous if-then rules which are applied to each document that must be classified [43]. For example, when

categorising legal texts into categories, rules can be made that see if a legal text contains words such as "criminal", "evidence", or "murder", which would then see the classification of the legal text to be labeled as "criminal law". However, a rule-based model can only be effective when there are rules for anything that may come up in the texts that need to be classified. In addition, the rules' conditionalities are based on domain knowledge which requires an expert in the domain to help with the crafting of said rules [25; 43].

2.3.1.2 Distance-based models

An example of a distance-based model is the k-nearest neighbors algorithm. This algorithm clusters the documents in k clusters based on the distance between points (i.e. documents) [43]. The distance between documents is calculated by looking at a similarity score of words contained within the documents. When this model is presented with a new document, it calculates which points and/or cluster it is closest to (i.e. distance function). In other words, it looks at how similar the presented documents are to the clusters that have been made. However, finding a meaningful distance function is deemed to be difficult for datasets consisting of only text [43].

2.3.1.3 Tree-based models

There exist a number of tree-based models with the primary ones being decision trees and random forests. These tree-based models are somewhat similar to rule-based models as they divide the predictor space into regions of choices (e.g. "Is the text longer than ten characters?"). As described by Kowsari, decision trees have been successful in a wide range of classification tasks [43]. Decision trees create nodes by splitting data into their children by fragmenting data with the help of constraints. Constraints can be binary, distinct values, or a combination of both.

Decision trees are very easy and fast to train but they are prone to overfitting [44] and have problems with out-of-sample prediction [45]. There are techniques to combat these phenomena, but there is no consensus on their adequacy [43; 44].

Random forests are an ensemble technique for classification tasks. This technique was first introduced by Ho (1995). This technique trains multiple decision trees randomly and picks the best-performing decision tree based on voting. Random forests are fast to train but they are slow with predicting once trained [43; 46].

2.3.2 Complex text classification methods

For complex text classification methods, probabilistic classifiers, Support Vector Machines (SVMs), and neural networks are considered.

2.3.2.1 Probabilistic

The NB classifier makes use of the Bayes' theorem of conditional probabilities [43]. For every feature in the text that is classified, the Bayes' Theorem calculates the probability for a class. This probability is dependent on the value of each feature. Each calculation for a feature's probability is done independently. The reason for this is the naive assumption of conditional independence between features. The NB classifier is a fast algorithm, but is limited when data are scarce since a likelihood value must be computed for every possible feature value in the feature space [43].

For this research, the multinomial variant of the Naive Bayes model is used. The multinomial variant takes word vectors as an input (either all words or a fractional counts such as Term Frequency - Inverse Document Frequency). For this variant, there are two most important hyperparameters to consider, namely α and a prior fit. The alpha parameter refers to the degree of smoothing performed on the maximum likelihood of a feature appearing in a sample belong to a specific class. Setting the alpha level below 1 is called Lidstone smoothing and setting the

alpha level at 1 is called Laplace smoothing. A prior fit refers to the classifier's ability to learn the class probabilities (i.e. class imbalance) prior to fitting.

2.3.2.2 Support Vector Machines

SVMs were first brought into the world by Cortes & Vapnik [47]. An SVM constructs one or more hyperplanes in a high-dimensional space which is used for ML tasks such as prediction or outlier detection. SVM joins a kernel technique with a framework that minimizes structural risk. The kernel function maps the features with a module across a suitable high-dimensional space and applies a learning algorithm that finds linear patterns across this new feature space. In a study performed by Goncalves & Quaresma multiple monolingual SVM classifiers were combined into a single framework that was able to replicate the performance of monolingual classifiers [48]. The downsides of an SVM come from memory complexity, a lack of transparency in results due to a high number of dimensions, and choosing the appropriate kernel is difficult [49].

For this research, the C-Support Vector Classification (SVC) variant of an SVM model is used. In practice, an SVC model supports different kernels (i.e. linear, poly, rbf, and sigmoid). However, due to the fact that the time to fit a model to the data scales at least quadratically with the number of samples, only a linear kernel is considered [49; 50].

Aside from the kernel that is used in an SVC classifier, there are two other hyperparameters that are essential for its performance, namely the C value and the class weights. The C value refers to the regularization parameter. It is a penalty for misclassifications. The higher the value, the more the classifier is penalised for wrongly classifying samples. However, the higher the bias, the higher the variance between predictions. Class weight refers to the weight attributed to a class. In imbalanced datasets, it is inversely proportional to class frequencies in the input

data. Meaning that if class weight is set at balanced, it treats all classes equal despite the discrepancy in class frequencies.

2.3.2.3 Neural networks

Neural networks consist of one or more input layers, hidden layers, and an output layer [25]. Each layer consists of nodes. Layers are connected through links between these nodes.

Such nodes are created like the neurons in a human brain. Nodes refer to the active number of units active in the neural network. These neurons are a set of incoming connections and outgoing connections where they apply weights and an activation function to the inputs so that the neural network can learn their importance. The number of neurons in a neural network is closely related to the network's proneness to under fitting or overfitting. The input layer is the layer that receives the inputs to the problem the neural network must solve. It feeds data into the network which the hidden layers pick up. Hidden layers are layers that are located between the input and output of the algorithm and perform complex, non-linear mathematical functions to convert an input to an output. Since hidden layers allow the neural network to perform complex mathematical functions, the more hidden layers, the more complex patterns the neural network can learn from the data.

These deep neural networks often experience overfitting [51]. To address this problem, a technique called dropout is used. Dropout refers to a regularisation method where some number of layer outputs are dropped out. The dropout's value determines the chance of this happening. This thins out the network in terms of predictive ability, as it forces the neural network to learn a sparse representation of the data. This is useful for when there is a small amount of data or if the neural network is large.

Convolutional Neural Networks (CNNs) are a type of neural network that are commonly used for image and video recognition tasks. CNNs are designed to automatically learn spatial hierarchies of features from raw input data, such as images, by using (convolutional) filters [52] The key idea behind CNNs is to exploit the spatial correlations in the input data by convolving small filters over the input to extract local features, and then pooling these features to create higher-level representations. The filters can be learned during the training process but can also be specified beforehand [52].

Recurrent Neural Networks (RNNs) are a type of neural network that is particularly well-suited for analysing sequential data, such as time series or natural language text. This type of neural network was first introduced by Rumelhart et al. (1985). Unlike feedforward neural networks, which process each input independently, RNNs maintain a "memory" of previous inputs, which allows them to capture the temporal dependencies in the data. The basic structure of an RNN consists of a single hidden layer of neurons, where each neuron is connected to both the input and output layers. The key feature of an RNN is that the output from each hidden neuron is fed back as input to the same neuron in the next time step, creating a "recurrent" loop [53].

Bi-directional Long Short-term Memory (BiLSTM) networks are a type of recurrent neural network (RNN) that has the ability to process sequential data in both forward and backward directions. This architecture was first introduced by Schuster & Paliwal (1997). It is comprised of two LSTM layers, one processing the input sequence in a forward direction and the other processing it in a backward direction. The output of the two LSTMs is then concatenated at each time step to produce the final output. They have been shown to be effective in NLP tasks including language modeling, translation, and text classification [54].

Transformer Models (TMs) were first introduced by Vaswani et al. [24].

TMs put aside a recurrent or convolutional nature of neural networks as they solely rely upon an attention mechanism for establishing global dependencies between inputs and outputs of the model. The attention mechanism is described as "mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key" [24]. By eschewing recurrence and convolutions, it is now possible to parallelise training as attention enables the model to perform multiple projections of d-dimensional keys, values, and queries. This is named multi-head attention. With multi-head attention, it allows the TM to learn long-range dependencies easier than RNNs or CNNs [55]. This is the case as a TM connects all positions with a constant number of sequentially executed operations, shortening the length of the path of forward and backward signals and thus shortening the paths between any combination of positions in the input and output sequences. Although TMs could theoretically handle arbitrarily long inputs, because memory usage scales quadratically with sequence length, an accepted input size is often 512 or 1024 tokens.

Nowadays, a new type of TM is considered the state-of-the-art in TM architecture, namely Bidirectional Encoder Representation Transformers (BERT) which was introduced by Devlin et al. (2018). BERT is adapted from TMs to allow unidirectionality. TMs were first constrained by having to move input tokens from the left-to-right inside its architecture. BERT allows the architecture to introduce Masked Language Modeling (MLM), which fuses contexts of both sides of the input by masking some tokens in pre-training and letting the model predict the original token. This way, BERT achieves state-of-the-art performance in eleven NLP tasks at the time of publication [56]. In practice, BERT models are pre-trained on millions of tokens in an unsupervised manner so that they are easily fine-tuned for a

downstream task (i.e. the task that is the actual objective of the model).

Multilingual models are an instance models where the model is pre-trained on multiple languages instead of one [18; 57]. Furthermore, multilingual models learn word representations of all the feature languages by encoding pairs of words that share the same meaning [56]. This allows the transfer of knowledge between languages. In practice, there are two major variants of BERT-based models for multilingual modeling, namely m-BERT and XLM-R.

m-BERT is trained on a large Wikipedia text corpus of the top 104 languages with an MLM objective [57].

XLM-R is an adaptation of BERT [18] and a new iteration of XLM [58]. XLM-R uses Byte-Pair Encoding (BPE) instead of using tokens as input. Additionally, XLM-R is pre-trained multilingually on 104 languages by having each training simple be paired with its translation in another language. This allows the MLM procedure to predict a token in another language while BERT is constrained to predicting masked tokens in the same language. In the introductory paper of XLM-R by Conneau et al. [18] it is stated that XLM-R outperforms m-BERT "on a variety of cross-lingual benchmarks, including +14.6% average accuracy on XNLI, +13% average F1 score on MLQA, and +2.4% F1 score on NER" at the time of publication.

2.3.3 Explainiable AI (XAI)

Artificial Intelligence (AI) is constantly pushing the boundaries of what we thought possible for machines and automation [59]. However, with their increased complexity comes a drawback, namely the lack of human understanding. When AI comes to impact our lives in healthcare, law, and policing, it is essential to start understanding how AI generates these decisions [**?**]. This striving for understanding spawned the field of XAI. This section discusses literature on the state-of-the-art

methods for XAI, touches upon the outstanding challenges, and reviews XAI for neural networks.

Arrieta et al. state the most commonly used nomenclature for evaluating the explainability of AI: [42]

- **Understandability** denotes the characteristic of a model to make a human understand its function without any need for explaining its internal structure or the way the model processes data internally.
- **Comprehensibility** refers to the ability of a learning algorithm to represents its learned knowledge in a human understandable fashion.
- **Interpretability** is defined as the ability to explain or to provide the meaning in understandable terms to a human.
- **Explainability** denotes how an explanation serves as an interface between humans and a decision maker.
- **Transparency** is concerned with the fact whether a model is understandable by itself.

Arrietta et al. define XAI as: *"Given an audience, an explainable Artificial Intelligence is one that produces details or reasons to make its functioning clear or easy to understand".*

Arrietta et al. distinguish between three types of XAI techniques for interpreting models, namely: model-agnostic post-hoc explainability techniques, post-hoc explainability for shallow ML models, and post-hoc explainability for deep learning models. Shallow ML models are defined as models that do not rely on layered structures of neural processing units. Therefore, deep learning models are defined as models that do rely on layered structures of neural processing units. Since

only model-agnostic post-hoc explainability techniques and post-hoc explainability techniques for deep learning models fit the scope of this research, only these techniques will be reviewed.

Post-hoc model-agnostic techniques can be applied to any model as they do not attempt to explain the inner workings of the model. Instead, model-agnostic techniques extract knowledge from models or reduce their complexity by visualization. Model-agnostic techniques consist of model simplification, feature relevance estimation (i.e. ranking the importance of model inputs for certain outputs), and visualisation techniques (e.g. visualising the, often simplified, structure of a model).

Post-hoc explainability techniques for deep learning models At the time of writing, Arrietta et al. did not considered writing about TMs or pre-trained LMs. Therefore, their overall recommendations for understanding deep learning models are considered.

In general, there are two paths that can be followed to understand deep learning models:

- Exampled-based explanation: An explanation that tries to detail the decision process of the model by mapping back the output in the input space to see which parts of the input were essential for the output.
- Model-based explanation: An explanation that tries to encapsulate the inner workings of the model albeit that no specific inputs are reviewed.

There are several ways one can approach both paths to understand deep learning models and more specifically, pre-trained LMs. First, one can leverage the capabilities of a XAI framework named Captum. Captum is a framework built for model interpretability built on PyTorch. There are three main features for model interpretability:

- Primary: Allows the attribution of output predictions of deep learning models to inputs. This can be used to understand how the model came to its predictions.
- Layer: Allows the attribution of output predictions to all neurons in a hidden layer. This can be used to assess what the contributions of each hidden layer are to a prediction.
- Neuron: Allows the attribution of an internal, hidden neuron to the input of the model. This can be used to assess how each neuron fires when it is provided with a given input.

Another possibility for understanding deep learning models, is BertViz [60]. BertViz is a tool that helps visualise self-attention in a BERT-based model. Similar to Captum, BertViz is capable of visualising three categories of a BERT-based model, namely the attention-head level, the model level, and the neuron level. The attention-head level visualises the patterns of attention heads per layer of the transformer model. The model level provides an overview of how the attention mechanism functions across the model's layers and heads. The neuron level visualises the individual neurons in the query and key vectors of each part of an input. It shows how the query, key and query x key interact with each other to generate the attention scores.

2.4 Threat Classification

Oostdijk & Van Halteren investigate to what extent threats can be detected in Dutch tweets using a definition that was constructed by reading Black Law's Dictionary [30] and the Canadian Criminal Code [61]. The authors train two classifiers: one classifier is trained on manually constructed n-gram patterns and the

other is an ML approach that uses n-grams as input. These two classifiers respectively achieve 79.9% and 90.1% precision on the test set (i.e. 10% of the total data).

Hammer et al. present THREAT, the largest publicly available dataset containing violent threats made online. The dataset is comprised of roughly 30.000 sentences from 10.000 YouTube comments which have been manually annotated for containing a violent threat or not [62]. 1,384 sentences were annotated for containing a violent threat. The dataset was first introduced by Hammer [9] but only made publicly available in 2019 [62]. However, Hammer et al. do not present a definition that is used for the annotation of said threats.

Wester et al. uses the non-public version of the THREAT corpus to train various classifiers based on enriching the corpus with linguistic features. These features are lexical and morphosyntactic information acquired through Part of Speech tagging and dependency parsing. The best performing classifier on a hold-out set was an SVM classifier relying on n-grams with an F-score of 0.6885 [63]. However, it is unknown which SVM variant was used and which kernel the authors decided to use.

Stenberg also uses the THREAT corpus to apply a convolutional neural network (CNN) classifier to the task of classifying threats and non-threats in this corpus [10]. Stenberg trains the CNN and applies different pre-trained word embeddings such as Wikipedia + Gigaword, Twitter, and Common Crawl. After hyperparameter tuning an F-score of 0.6237 is achieved.

Ashraf et al. also use the THREAT corpus with the goal of differentiating when a threat is targeting individuals or a group [64]. The authors augment the dataset by adding labels that indicate whether an individual or a group is targeted by a threat. Afterwards, they compare deep-learning classifiers, namely: LSTM, BiLSTM, and a CNN classifier. These classifiers respectively achieved F1-scores

& ROC-AUC scores of 0.83 & 0.93, 0.85 & 0.94, and 0.82 & 0.93. The authors argue that pre-trained models do not work as well due to them being trained on a different social media platform than the origin of the THREAT corpus.

AlAjlan & Saudagar investigate Arabic comments on Instagram posts to detect threats [8]. The authors train a CNN classifier to detect manually labeled data. The data consist of 2,000 Arabic comments comprised of 1,000 threats and 1,000 non-threats. Their CNN model was reported to achieve an accuracy of 99% and an F1-score of 0.99 for predicting labels for threats and non-threats. The authors do not state annotation guidelines for reproducibility. AlAjlan & Saudagar also do not present a definition used for what constitutes a threat and what does not.

Chakraborty & Seddiqui (2019) train two ML classifiers (i.e. SVM & Multinomial Naive Bayes) and one deep-learning classifier (i.e. CNN with LSTM) on Facebook comments containing abusive language in Bengali [6]. The collected dataset is balanced due to there being a 50-50 split in either abusive or non-abusive language. For the SVM, MNB, and CNN classifiers accuracies of 78%, 70%, and 77,5% were obtained. Chakraborty & Seddiqui also do not present a definition used to annotate their data for threats and non-threats.

2.5 Summary

This chapter has given a thorough review of existing literature in the fields of threats, text classification, and online threat classification. It was found that threats can be a variety of things, for example, an expression of anger or humor. A threat can be distinguished from other illocutionary acts by reviewing three parameters:

- C1. The speaker's intention to personally commit an act (or be responsible for bringing about the commission of the act);

- C2. The speaker's belief that this act will result in an unfavorable state of the world for the addressee.
- C3. The speaker's intention to intimidate the addressee through the addressee's awareness of the intention in C1.

It is uncertain whether textual features alone can distinguish threats from non-threats. Fraser argues that is probably impossible to make a context-independent assessment of this [28]. However, in a study performed by Nini, it was found that it is possible to discern this.

Furthermore, this chapter examined text classification methods and explainability of deep-learning models, also in the context of threat classification. It was found that there are simple and complex text classification methods that each have their own pros and cons. Simple text classification methods are simple to train but are often computationally expensive to train, are prone to overfitting, and lackluster in predicting out-of-sample. Complex text classification methods are often limited by data scarcity, hard to interpret, and can have limited input size issues. However, pre-trained language models can be used to form the foundation of a model which can be fine-tuned on a downstream task. This approach has been shown to deliver state-of-the-art results in the majority of NLP tasks. Pre-trained language models can be monolingual or multilingual. The two state-of-the-art architectures for this are BERT and XLM-R. To evaluate these complex architectures, feature relevance estimation is used to decompose the neural network into the importance of input features.

When reviewing the existing literature on threat classification, it can be concluded that the results cannot be generalised due to an inconsistent definition of illegal threats. Furthermore, there are no studies that have leveraged the new technology of pre-trained models. The majority of these studies have used other types of algorithms, namely CNNs, SVMs, LSTMs, and Naive Bayes.

Chapter 3

Datasets

The datasets used in this research come from different sources and are diverse in nature. For both languages, Dutch and English, an overview of the data's metadata will be provided.

3.1 Dutch dataset

The Dutch dataset is constructed by scraping alleged death threat tweets from Twitter and supplementing it with non-threats which are also scraped from Twitter. To ensure a realistic class imbalance for a real world scenario (as requested by the NP), a class imbalance of 1/100 is chosen with threats being the minority class. This class imbalance was deemed realistic as this reflects the class imbalance of data collected by the NP in a similar dataset. It should be noted that for the Dutch death threat tweets, the number of alleged death threats will vary after the annotation task has been completed. It is expected that the number of alleged death threats is significantly higher than the resulting number of LATs as an output of the annotation task. To ensure classifier robustness, the texts that are annotated as non-threats are kept in the dataset.

3.1.1 Dutch death threat tweets

A civilian initiative founded the website `www.doodsbedreiging.nl`. It warehouses thousands of death threats in Dutch that were sent through Twitter by other people. The initiative allegedly wanted to raise awareness for death threats being made in the online domain. Since the website is no longer online and the tweets are therefore no longer accessible through this channel, the *doodsbedreiging* Twitter account was scraped to download the tweets.

A total of 5,098 tweets were scraped from the *doodsbedreiging* Twitter account[1]. The oldest dates back to 2011 and the newest tweet dates back to 2017.

3.1.2 Dutch non-threats supplementation

To supplement the Dutch tweets with non-threats, another dataset is used. To ensure that the supplemental data are comparable to the death threat tweets, a similar source must be chosen. The reason for this is to prevent spurious signals dictating the classification (e.g. text length and vocabulary). Therefore, Dutch tweets are scraped from the Twitter platform that are comparable to the death threat tweets in terms of text length. Since a class imbalance of 1/100 is chosen for this research, a total of 504,702 Dutch non-threats are scraped from Twitter.

3.1.3 Descriptive statistics on Dutch dataset

To get a sense of the data and by means of exploratory data analysis, descriptive statistics are provided. The descriptive statistics of the internal police data are presented in Table 1. The distribution of text lengths is presented in Figure 1. It must be noted that statistics regarding text length indicate the amount of characters a text consists of.

[1] www.twitter.com/doodsbedreiging

Statistic	Value
Nr. of samples	500577
Majority class samples	498232
Minority class samples	2325
Mean text length	47
Median text length	204
Minimum text length	1
Maximum text length	333

Table 1: Descriptive statistics on the Dutch threat corpus

3.2 Internal police data

The supervisor at the NP granted me indirect access to a confidential dataset. This dataset is comprised of Dutch telecommunications (i.e. texts) between humans. The form of the texts is best described as short e-mails.

3.2.1 Descriptive statistics on internal police data

To get a sense of the data and by means of exploratory data analysis, descriptive statistics are provided. These statistics can be compared to those of the constructed Dutch corpus detailed in Section 3.1.3. The descriptive statistics of the internal police data are presented in Table 2. The distribution of text lengths is presented in Figure 2. It must be noted that statistics regarding text length indicate the amount of characters a text consists of.

3.3 English datasets

The English dataset is constructed from multiple sources that feature texts that are labeled to contain a threat or non-threats. From each dataset mentioned below, the alleged threats are extracted. Afterwards, these threats are supplemented with

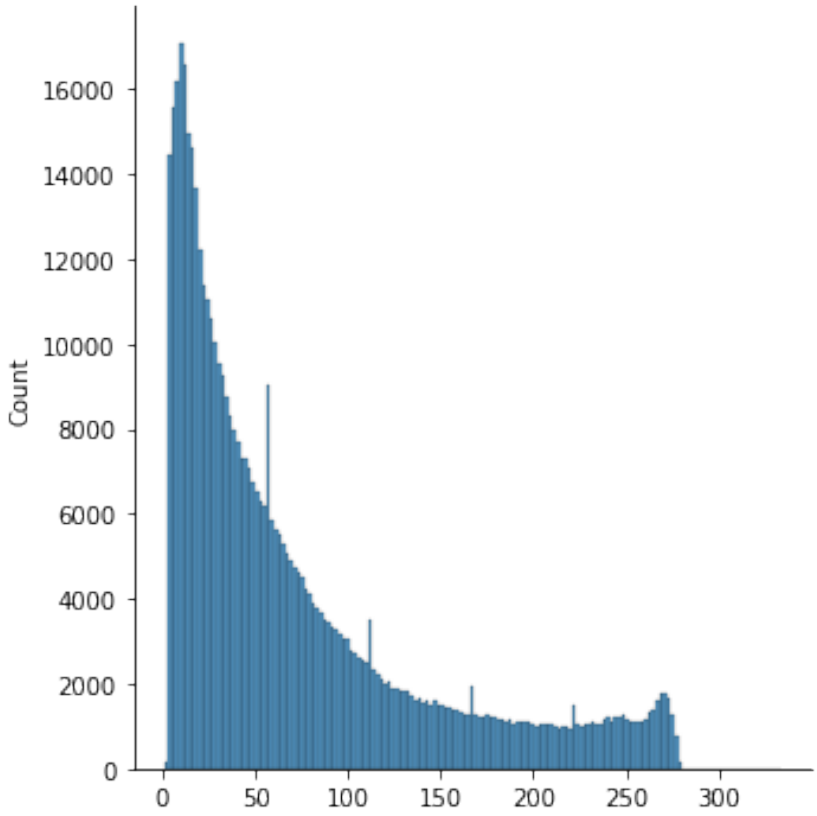

Figure 1: Text length distribution of the Dutch threat corpus

Statistic	**Value**
Nr. of samples	2199
Majority class samples	2197
Minority class samples	2
Mean text length	478
Median text length	204
Minimum text length	3
Maximum text length	18337

Table 2: Descriptive statistics on internal police data

non-threats to achieve the chosen class imbalance of 1/100 with threats being the minority class.

It should be noted that for the English threats, the number of alleged threats

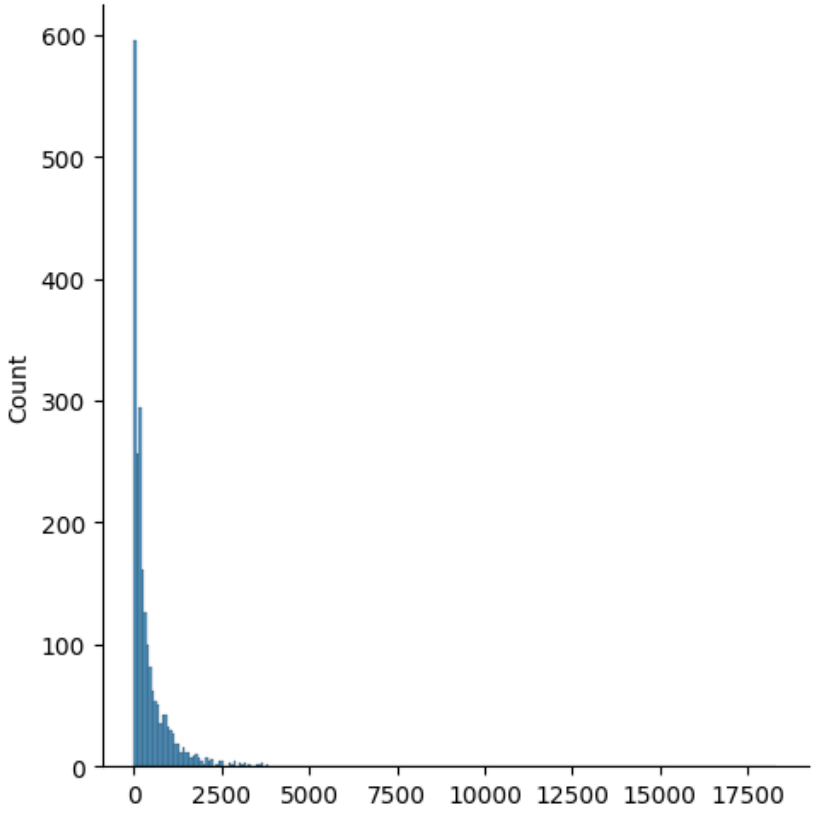

Figure 2: Text length distribution of the internal police data

will vary after the annotation task has been completed. It is expected that the number of alleged threats is significantly higher than the resulting number of LATs as an output of the annotation task. To ensure classifier robustness, the texts that are annotated as non-threats are kept in the dataset.

3.3.1 English threats

3.3.1.1 Jigsaw Toxic Comment Classification Dataset

In a Kaggle competition by the Jigsaw team of Google a dataset is shared[1]. This dataset can be used for classifying toxicity for online comments on Wikipedia.

[1]https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

This dataset is comprised of 150000+ Wikipedia comments. This dataset features 478 comments annotated as a threat. The annotation process was performed by crowd-evaluation.

3.3.1.2 Jigsaw Unintended Bias in Toxicity Classification Dataset

In a Kaggle competition by the Jigsaw team of Google a dataset is shared[1]. This dataset can be used for identifying unintended bias in toxic comments online. This dataset features 148 comments annotated as a threat.

3.3.1.3 THREAT: Violent threats on YouTube

Hammer et al. (2019) present THREAT, the largest publicly available dataset containing violent threats made online. The dataset is comprised of roughly 30000 sentences from 10000 YouTube Comments which have been manually annotated for containing a violent threat or not [62]. 1384 sentences were annotated for containing a violent threat. The dataset was first introduced in 2014 by Hammer (2014) [9], but was only made publicly available in 2019 [62].

3.3.1.4 Directo Lim corpus

Directo Lim (2018) created one dataset[2] from a portion of a dataset published by Davidson et al. (2017) [65]. The Directo Lim corpus is a collection of 24782 tweets that contain terms from a hatespeech lexicon (i.e. words and phrases that have been identified by internet users as hate speech) compiled by `www.hatebase.org`. These tweets were manually annotated by CrowdFlower workers. Directo Lim's dataset was constructed by manually going through this dataset and annotating the rows where the tweet was labeled as hate speech. This resulted in 59 threats

[1] https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification
[2] https://drive.google.com/open?id=1wDJTTwbsKZLsny7UDYberHaj356niulc

being labelled as a threat and 2676 non-threats. These tweets were labeled by two licensed lawyers in the Philippines with experience in criminal law. There was moderate agreement between the two annotators (Cohen's Kappa = 0.46).

3.3.1.5 Contextual Abuse Dataset (CAD)

Vidgen et al. (2020) present the CAD, a dataset comprised of roughly 25000 Reddit entries[1]. The data were manually annotated using salient subcategories. The subcategory relevant for this study is *Threatening language*. Filtering out data entries with this subcategory, 34 threats were extracted to be used in this study.

3.3.2 English non-threats supplementation

To supplement the English threats with non-threats for classifier robustness and generalisability, two other datasets are used. Firstly, a random sample is taken from the dataset mentioned in Section 3.3.1.1. This sample is taken from the test set which does not overlap with the sample taken in Section 3.1.2. Secondly, a sample is taken from Twitter by scraping the platform for English tweets.

For this research, a class imbalance of 1/100 is chosen. One percent are threats and 99 percent are non-threats. This means that 100485 texts are sampled from the dataset detailed in Section 3.3.1.1 and 100485 texts are scraped from Twitter by scraping the platform for English tweets.

3.3.3 Descriptive statistics on English dataset

To get a sense of the data and by means of exploratory data analysis, descriptive statistics are provided. The descriptive statistics of the internal police data are

[1]https://github.com/dongpng/cad$_n$*aacl*2021

presented in Table 3. The distribution of text lengths is presented in Figure 3. It must be noted that statistics regarding text length indicate the amount of characters a text consists of.

Statistic	**Value**
Nr. of samples	9401
Majority class samples	9013
Minority class samples	388
Mean text length	193
Median text length	84
Minimum text length	3
Maximum text length	4829

Table 3: Descriptive statistics on the English threat corpus

3.3.4 Data pre-processing

Since all texts come from social networks, they must be pre-processed to achieve a uniform type of text. This section highlights which parts of the texts are removed or changed to achieve this.

3.3.4.1 Removing parts of texts

For tweets, it is possible that tweets are re-tweeted, meaning that a pre-fix "RT: " is added. This pre-fix is removed from all texts acquired from Twitter. Additionally, hashtags and punctuation marks are removed completely, aside from the at sign (i.e. @) which is explained in the following section.

For all texts, numbers are removed from the texts as they do not form a meaningful element of a potential threat.

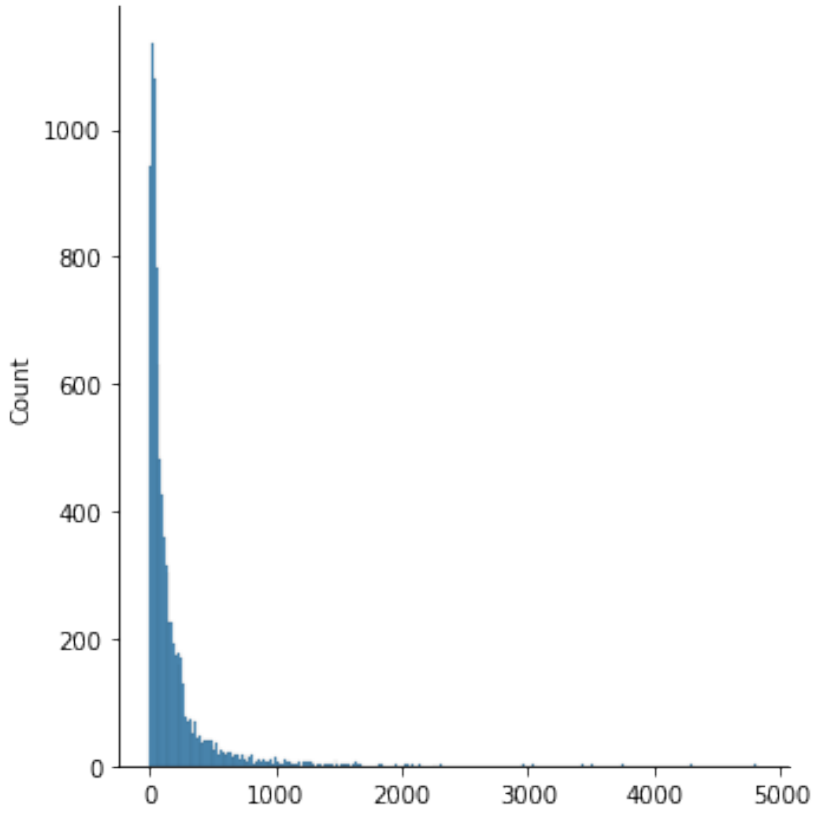

Figure 3: Text length distribution of the English threat corpus

3.3.4.2 Changing parts of texts

For tweets, it is possible that people refer to other people by mentioning their Twitter handle (e.g. "Hey @DonaldTrump"). If this handle is completely removed, important information is lost as a person is addressed which is an important element of a threat. Therefore, the "@" is removed from mentions.

For all texts, all letters are made lowercase to artificially reduce the vocabulary size of the texts.

3.3.5 Summary

This chapter has given an overview of the collected data that is both in English and in Dutch. For each language, the respective datasets are reviewed.

For the Dutch language, a total of 509800 tweets are collected that contain 5098 alleged death threats. These data are gathered from one source, namely a civilian initiative called doodsbedreiging on Twitter.

For the internal police data, a total of 2199 samples are collected that contain 2 LATs. This data is best described as telecommunications between humans that resemble e-mails.

For the English language, a total of 246300 texts are collected that contain 2030 alleged threats. These data are gathered from five sources. Three of these five sources' data focus on abuse or toxic comments and two of these five sources' data focus on threats.

For the alleged threats of both languages, the annotation task must be completed to verify if these alleged threats are legally actionable. It is expected that the number of alleged threats is significantly higher than the resulting number of LATs as an output of the annotation task. To ensure classifier robustness, the texts that are annotated as non-threats are kept in their respective dataset.

Chapter 4

Methodology

This section outlines the methodology used to investigate how the legal definition of a threat can be learned programmatically. The foundation for this methodology is adapted from the Design Cycle from Wieringa [66]. The relevant sections of Wieringa's Design Cycle for this research pertain to a problem investigation, treatment design, and treatment validation. A problem investigation explains the problem that is experienced at the target institution. A treatment design refers to the designing of a technique, method, algorithm or conceptual framework that is applied to the problem that was identified during the problem investigation. Afterward, during the treatment validation, the treatment is validated through different means of statistical measurement.

4.1 Problem investigation

The problem context of this thesis is the need of the NP to survey online discourse and assess if potentially dangerous situations can arise for a person or the public order. Examples of this are people threatening to commit a crime at high-risk events such as large-scale demonstrations, football games, or concerts. Another

example of this is people threatening other people on social media. To assess these potentially dangerous situations, the NP monitor online discourse on social networks. The online discourse that is monitored can be upwards of millions of messages. Therefore, it is imperative to filter out messages which are most likely non-threatening and retain as much messages that could be (legally actionable) threats. Furthermore, online discourse is monitored in multiple languages with Dutch and English being the most important languages.

The stakeholders for this research are the Data Intelligence Team of the NP. They want to filter millions of messages of social networks to classify as many potential threats for high-risk events so that they can manually investigate them. Therefore, the scope of this research is limited to the Dutch judicial system where algorithms are used to provide the police with insights. Moreover, a classification model should be used as an addition to the expert judgment of police employees because assessing potential threats is too time-consuming and solely relying on a classification model is unwise. Furthermore, since these types of models are not widespread or the norm for the police, it is crucial that these models are explainable to prevent biases and investigate wrong classifications. Wrong classifications, especially false negatives, can be detrimental to the work of the police. Investigating classifications requires a lot of time and having many wrong classifications impairs the work of the police as they are not able to spend time on more credible threats.

Another problem the NP is experiencing is data scarcity. There are not enough data to build a reliable model on. However, the NP can provide a small test set. Therefore, a classification model ought to be built with annotated open data so that this model can be tested on unseen data of the police.

The literature review in this study is performed as part of the problem investigation. It further investigates the current problems researchers are having

classifying threats and exposes gaps in the literature that this research will address. These findings form input for the subsequent designs of treatments and their validation that are performed in this research.

4.2 Treatment design

Wieringa (2014) defines a treatment as consisting of an artifact interacting with a problem context. In this research, the artifact consists of a set of Python scripts that ultimately create models which are derived from texts that serve as training data. Therefore, the problem context consists of a need to construct a Dutch and English corpus that are constructed from online sources that contain alleged threats. To ensure a consistent definition for a threat is used, an annotation guideline must be created. This annotation guidelines must be applied to the found texts containing alleged threats. To investigate the adequacy of the language model classifier's performance, it must be compared to benchmark models (which were defined in 1). Thus, the goal of the interactions of the artifacts (i.e. classification models) with the problem contexts is to predict if texts can be accurately classified as containing legally actionable threats or not using the Dutch legal definition as applied with a set of annotation guidelines.

For both the Dutch and English corpora, alleged threats are manually re-annotated for containing legally actionable threats. The data that are re-annotated are defined in Section 3.1.1 and Section 3.3.1. For the annotation task, two people (i.e. the researcher and a willing participant) annotate using annotation guidelines. The annotation guidelines inform the annotators how to use the Dutch legal definition to gauge whether a threat can be considered legally actionable. An annotation is completed when an annotator has indicated if a text contains a legally actionable threat and if so, which part of the text signifies this. The annotation

guidelines can be observed in Appendix B.

To gauge the reliability of annotations, Cohen's Kappa is calculated. This score measures the agreement between rates who each classify n items into c mutually exclusive categories. In this case, there are two raters and two categories (i.e. LAT and non-threat). The second annotator was not able to manually re-annotate roughly 7500 texts. Therefore, to have a Cohen's Kappa score that is within $\pm 5\%$ of the measured Cohen's Kappa value, a minimum number of overlapping texts must be re-annotated by the raters. It is then desirable to have a tight confidence interval (i.e. a high chance that the measured Cohen's Kappa is within the calculated interval of the actual Cohen's Kappa). To calculate this minimum number of overlapping texts, the following formula is used:

$$Minimum\ sample\ size = \frac{z^2\ *\ \sigma(1-\sigma)}{m^2}$$

With z being the desired confidence interval, σ being the standard deviation, and m being the margin of error. The formula for the z-score is as follows:

$$z = (x\ -\ \mu)/\ \sigma$$

When calculating the required sample size with a confidence interval of 95%, a margin of error of 5%, a population of 7500, and a standard deviation of 50%, the resulting minimum number of samples is 366.

For the training, validation, and testing of each treatment (i.e. the language model & benchmark models), a stratified 10-fold strategy is used. This means that ten different sets (i.e. folds) of training, validation, and testing data are created. For each fold, models are trained on the corresponding training data, models' hyperparameters are tuned on the corresponding validation data, and models' performance metrics are calculated on the corresponding testing data. This way,

5 x 10 = 50 models were trained instead of 5 (per corpus). These models are then compared by means of statistical measurement which is explained in Section 4.3.

4.2.1 Hyperparameter tuning

4.2.1.1 BiLSTM classifier

The hyperparameters of the BiLSTM classifier that must be tuned and their possible values are:

- Dropout: [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
- Recurrent dropout [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
- Hidden layers [0, 1, 2, 3]
- Nodes [2, 4, 8, 16, 32]
- Optimizer [Adam, SGD]
- Word embedding dimension: [50, 100]
- Sequence length [50, 75, 100]

As explained in Section 2.3.2, dropout refers to a regularisation method where some number of layer outputs are dropped out. The chance of this happening ranges from [0, 1]. Therefore, all possible values are tested for this hyperparameter in increments of 0.1.

The number of hidden layers in a neural network influences the neural network's ability to solve complex, non-linear problems. However, not all problems are created equal. Meaning, that a neural network does not always perform better with an increased number of hidden layers as less complex problems often require fewer hidden layers [67]. In a study performed by Karsoliya, it is theoretically

proven that adding four or more hidden layers to a neural network may lead to to problems in the training phase [68]. Additionally, Karsoliya argues that most problems that can be solved by neural networks require zero to three hidden layers. Therefore, this is the range for the hyperparameter that is tuned.

In the scientific literature, it is still a point of contention on how the optimal number of neurons in a neural network are determined. In a literature review performed by [69], it is concluded that many studies implemented different strategies for estimating the optimal number of neurons in a neural network when optimizing for efficiency, minimal errors, and accuracy. However, it is an essential step in the process of optimizing a neural network's performance. In the study performed by Karsoliya, three rules of thumb are used as a starting point for determining the number of neurons in each hidden layer, namely:

- The number of hidden layer neurons are 2/3 (or 70% to 90%) of the size of the input layer. If this is insufficient then number of output layer neurons can be added later on.
- The number of hidden layer neurons should be less than twice of the number of neurons in input layer.
- The size of the hidden layer neurons is between the input layer size and the output layer size.

Using these as a starting point and iteratively adding neurons when performance lacking, the range that is given above was constructed.

An optimizer influences the neural network's learning rate and weights attributed to inputs. It refers to the use of a particular algorithm to control these variables. Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. SGD is a gradient descent (with momentum) optimizer.

The word embedding dimension refers to the number of dimensions in a space where each token in the vocabulary is represented by a real valued vector. Tokens that are learned to be similar will be closer to each other in the vector space. The word embeddings that are used for training the BiLSTM classifier are the pre-trained GloVe Twitter data [70].

Sequence length refers to the maximum length of a text that can be used as an input.

4.2.1.2 CNN classifier

The hyperparameters of the CNN classifier that must be tuned and their possible values are:

- Maximum features (i.e. unique tokens): [10000, 20000, 33000, 50000]
- Number of filters: [32, 64, 128, 256]
- Word embedding dimension: [50, 100, 200, 300]
- Sequence length: [50, 75, 100, 125]
- Learning rate optimizer: [Adam, SGD]
- Learning rate scheduler: [Exponential decay, Reduce LR on Plateau, Linear Decay]
- Initial learning rate: [1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7]
- Learning rate decay rate: [0.5, 0.6, 0.7, 0.8, 0.9]

The number of maximum features refers to the number of unique tokens (i.e. words) in a text that will be used in the neural network. It is often beneficial to only use the most prominently used words or rather negate the use of words that are only

used once. The number of unique tokens in the Dutch corpus was 1.2 million and in the English corpus it was 0.4 million. Since increasing the maximum number of features for the CNN classifier drastically increases the training time, a maximum was set at 50000. This maximum still made it feasible to train the CNN classifier in a reasonable amount of time. The chosen values for this hyperparameter were done iteratively to optimize for training time.

The number of filters refers to a set of weights which is applied to any input passing through the neural network. When applied to the inputs, a feature map is created that summarises the existence of the features in an input. In a study by Ahmed & Karim, the impact of the number of filters is investigated. In their research, they argue that 64 or 128 filters are sufficient for most problems [71] To cover the most bases while maintaining a reasonable training time, this range is extended to [32, 64, 128, 256] as Stenberg uses 32 filters for his research on threat classification and 256 is seen as the upper limit for the number of filters [71].

The word embedding dimension refers to the number of dimensions in a space where each token in the vocabulary is represented by a real valued vector. Tokens that are learned to be similar will be closer to each other in the vector space. Since the GloVe Twitter embeddings are used, they only support dimensions of 50, 100, 200, and 300. Therefore, these values are also used for optimizing this hyperparameter.

Sequence length refers to the maximum length of a text that can be used as an input. As detailed in Section 3, the mean text length of the Dutch and English corpus is 47 and 193, respectively. Since the CNN classifier only considers a minimum of 10000 features and a maximum of 50000 features, it is expected that each text consists of significantly fewer tokens. Therefore, a range of [50, 75, 100, 125] is specified where texts with fewer tokens than 50 are padded to the specified sequence length.

An optimizer influences the neural network's learning rate and weights attributed to inputs. It refers to the use of a particular algorithm to control these variables. Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. SGD is a gradient descent (with momentum) optimizer.

A learning rate scheduler dictates the schedule on which the learning rate is updated at the end of each step in the training process. It can take the form of a logarithmic scale, linear scale, et cetera.

The initial learning rate refers to the speed by which the neural network learns patterns in the data. The higher the initial learning rate, the faster but more unstable the training process becomes. The slower the initial learning rate, the probability of failure to learn becomes more apparent. However, Mishra & Sarawadekar argue that no single policy on initial learning rate (and corresponding learning rate scheduling) can exist which is universally applicable [72] Therefore, learning rates from their experiments are used as a starting point (0.01 - 0.001) and added upon iteratively during the training process.

The learning rate decay rate sets the rate by which the learning rate scheduler decreases the learning rate. The suggested learning rate decay schedule is taken from the work from [72]

4.2.1.3 BERT classifier

The architecture from Delobelle, Winters, & Berendt (2020) will be used [73] for the language model's architecture. This architecture is based on the RobBertA architecture as proposed by Liu et al. (2019) [74]. This architecture was adopted by Delobelle, Winters, & Berendt to train a pre-trained language model with Dutch data and a Dutch tokenizer. The model was trained with a 39GB corpus consisting of 6.6 billion words in 126 million lines of text.

The hyperparameters of the BERT classifier that must be tuned and their possible values are:

- Initial learning rate: [1e-5, 2e-5, 3e-5, 4e-5, 5e-5, 5e-6, 5e-7]
-
- Learning rate decay rate: [0.5, 0.6, 0.7, 0.8, 0.9]
- Warm-up steps: [250, 500, 750, 1000]
- Batch size [4, 8, 16, 32]
- Gradient accumulation steps [4, 8, 16]

The initial learning rate refers to the speed by which the neural network learns patterns in the data. The higher the initial learning rate, the faster but more unstable the training process becomes. The slower the initial learning rate, the probability of failure to learn becomes more apparent. However, Mishra & Sarawadekar argue that no single policy on initial learning rate (and corresponding learning rate scheduling) can exist which is universally applicable [72] Therefore, learning rates from their experiments are used as a starting point (0.01 - 0.001) and added upon iteratively during the training process.

The learning rate decay rate sets the rate by which the learning rate scheduler decreases the learning rate. The suggested learning rate decay schedule is taken from the work from [72]

Warm-up steps refer to the number of steps it takes before the learning rate is adjusted. In practice, the initial learning rate is very low to allow the neural network to first make sense of the data before tuning the learning rate during the training process.

Batch size refers to the number of samples a neural network is fed at a time to learn the patterns in the data. After each batch, the optimizer steps and the learning rate is adjusted.

Gradient accumulation steps refer to the number of steps that the optimizer does not update the learning rate so that it can accumulate the gradients of the steps the model skips. Gradient refers to the calculation of how wrong the neural network was when evaluating what is has learned on a sample in a batch. By setting gradient accumulation steps, it is possible to split up batches into mini-batches and learn during a batch, rather than after.

4.2.1.4 Naive Bayes classifier

The choice was made to pursue the multinomial variant of the Naive Bayes (MNB) classifier rather than other variants (e.g. Complement Naive Bayes) due to its higher perceived performance before hyperparameter finetuning. This was tested by performing a single run on the data, without stratified k-fold splits. This resulted in other variants of the NB classifier still overfitting on the data despite simple adjustments to the hyperparameters. The MNB variant will be used in conjunction with Term Frequency - Inverse Document Frequency (TF-IDF). The hyperparameters of the MNB classifier that must be tuned and their possible values are:

- Alpha: [0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]
- Prior fit: [True, False]

The alpha parameter, also known as Laplace smoothing or Lidstone smoothing, is a hyperparameter that is used to add a small smoothing constant to the frequency count of each feature in the training set. This helps to avoid zero-frequency

problems and allows for more robust probability estimates. The value of alpha determines the strength of the smoothing. A larger value of alpha corresponds to stronger smoothing, and a smaller value corresponds to weaker smoothing. The optimal value of alpha depends on the size of the training set and the characteristics of the data.

The prior fit parameter refers to the classifier's ability to learn the class probabilities (i.e. class imbalance) prior to fitting.

4.2.1.5 SVM classifier

The choice was made to pursue the C-Support Vector Classification (SVC) variant due to its higher perceived performance before hyperparameter finetuning. This was tested by performing a single run on the data, without stratified k-fold splits. This resulted in other variants of the SVM (i.e. SVM, NuSVC, and LinearSVC) classifier still overfitting on the data despite simple adjustments to the hyperparameters.

Aside from that, the SVC classifier was pursued with a linear kernel instead of *poly, rbf,* or *sigmoid* since the time to fit a model scales quadratically with the number of samples. When other kernels besides the linear kernel were used, it would become impractical to train a model as it would be too time-consuming.

The hyperparameters of the SVC classifier that must be tuned and their possible values are:

- C: [1, 5, 10, 25]
- Class weight: [None, balanced]

The C value refers to the regularisation parameter. It is a penalty for misclassifications. The higher the value, the more the classifier is penalised for wrongly

classifying samples. However, the higher the bias, the higher the variance between predictions.

The class weight parameter refers to the weight attributed to a class. In imbalanced datasets, it is inversely proportional to class frequencies in the input data. Meaning that if class weight is set at balanced, it treats all classes equal despite the discrepancy in class frequencies.

4.3 Treatment validation

This section introduces the architecture of how the treatments are validated.

After creating five treatments for both Dutch and English, they will be validated against test sets in their respective language.

4.3.1 Validating Dutch treatments

The five treatments trained on Dutch data as defined in Section 3.1 are verified on two test sets. These test sets are named $TS_{D\text{-}O}$ and $TS_{D\text{-}NP}$. $TS_{D\text{-}O}$ refers to the Dutch data that is gathered from online sources as described in 3.1.1. $TS_{D\text{-}NP}$ refers to the Dutch data that is acquired from the NP as described in 3.2.

$TS_{D\text{-}S}$ is comprised of 20% of the Dutch data as defined in Section 3.1. This test sets contains both threats and non-threats. The descriptive statistics on this can be seen in Section 3.1.3.

$TS_{D\text{-}NP}$ is comprised of 2199 samples of internal NP data in Dutch containing both non-threats and LATs. The descriptive statistics on this can be seen in Section 3.2.

4.3.2 Validating English treatments

The five treatments trained on English data as defined in Section 3.3 are verified on two test sets. These test sets are named $TS_{E\text{-}O}$ and $TS_{E\text{-}NP}$.

$TS_{E\text{-}S}$ is comprised of 20% of the Dutch data as defined in Section 3.3. This test sets contains both threats and non-threats.

4.3.3 Performance metrics

Each treatment's application to the respective dataset yields performance measures for the testing scenarios. These performance measures are the F1-score and the PR-AUC. By comparing the PR-AUC and F1-scores between treatments for both $TS_{D\text{-}S}$ and $TS_{E\text{-}S}$, it allows the answering of RQ_2. By comparing the PR-AUC and F1-scores between treatments for both $TS_{D\text{-}NP}$ and $TS_{E\text{-}NP}$, it allows the answering of RQ_3.

4.3.3.1 F1-score

To account for imbalanced datasets where accuracy and recall are easily skewed, F1-scores are used. F1-scores are the harmonic mean of precision and recall. Precision refers to the classifier's performance in terms of the ratio of positive samples to all positive samples, predicted and actual. Intuitively, the precision score denotes the classifier's ability to not label a sample that is negative as positive. Recall refers to the classifier's performance in terms of the ratio of positive samples to the positive samples and falsely predicted positive samples. Intuitively, the recall scores denotes the classifier's ability to correctly classify the positive samples.

F1-scores entail the macro F1-score, the micro F1-score, and an F1-score per class in classification problems. The macro F1-score is calculated by taking the average of the individual F1-score of each class. The micro F1-score is calculated

by globally summing up all the FP, FN, TP, and TN counts for each of the classes and then calculating the F1-score using those global scores. For these reasons, the macro F1-score gives an equal weight to all the classes while the micro F1-score gives an equal weight to all the data points.

The formulae used for computing precision, recall, and the individual F1-scores are highlighted below.

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

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

$$F1_class = \frac{2\ *\ Precision\ *\ Recall}{Precision\ +\ Recall}$$

$$MacroF1 = \frac{sum(F1 - scores)}{numberofclasses}$$

$$MicroF1 = \frac{TP}{TP\ + 0.5\ *\ (FP\ +\ FN)}$$

4.3.3.2 PR-AUC

In addition to the F1-scores that will be reported, PR-AUC scores will be calculated. This is done by setting different classification thresholds in the range [0, 1] and converting class probabilities to binary predictions. For each of those classification thresholds, the precision and recall scores are calculated on the binary predictions. The result is a plot of precision versus recall, and the integral of this curve is the PR-AUC. The primary advantage of this approach is that the PR-AUC score does not exaggerate classifier performance for unbalanced datasets

[75].

4.3.4 Statistical testing of performance metrics

The comparison of the acquired performance metrics is done using statistical measurements. This section presents the evaluation strategy that is applied to compare performance metrics and the chosen statistical test.

For the evaluation of the models, the PR-AUC, the macro, micro, and per class F1-score are calculated for each fold. Additionally, a PR-AUC graph is plotted for each classifier based on the predictions on the fold that best represents the ten folds. This approach is chosen as it is impractical to plot ten PR-AUC plots in one figure while still being able to provide a graphicical representation of this performance metric. Additionally, a boxplot of the PR-AUC scores per fold is presented to highlight the PR-AUC scores' spread per classifier.

To test for a significant difference between the classifiers for each performance metric, a statistical test must be performed. For each statistical test, the data must meet certain assumptions in order to be considered a valid test for its data. Initially, a McNemar test [76] was considered. The following assumptions are made of the data:

- The dependent variable must be binary and dichotomous. This means that the dependent variable must place all samples into one group or the other.
- The pairs of data that are being tested on are dependent observations.
- The class distributions of the two samples should be equal.
- The samples are randomly collected from the population.

Since McNemar's test assumes that the class distribution of the two samples are equal, this test might not be valid as a way of testing the statistical significance

for the performance metrics of multiple classifiers. The reason for this is that the class distribution of the data is highly imbalanced.

Another way of testing the statistical significance for paired data is the (student's) t-test. The t-test makes the following assumptions for the data:

- The data are continuous.
- The samples have been randomly collected from the population.
- There is homogeneity of variance.
- The sample distribution is approximately normal.

The data that are collected for this research do not fit the assumptions made by the t-test. The data are binary, not continuous. There is no homogeneity of variance as the datasets are highly imbalanced. It is impossible to assume that the sample distribution is normal.

A Tukey Honestly Significant Difference (HSD) test can also be used to test differences among sample means for significance. This test has the following assumptions regarding the data:

- All observations are randomly and independently sampled
- The sample distribution is distributed normally
- The samples all have the same variance

These data assumptions can be met as a stratified 10-fold split allows each split to have a random sample and can make sure each split has the same distribution of classes. In practice, the Tukey HSD test is performed on the mean values of the classifier's performance on all 10 folds.

4.3.5 Feature relevance estimation

Lastly, feature relevance estimation is performed to assess which words of texts contribute towards a right or wrong classification of a non-threat or legally actionable threat. By then comparing this to the annotations of what parts of texts signify legally actionable threats, it allows the answering of RQ_4.

Using the feature relevance estimation module from the Captum framework for interpreting deep learning models, it was possible to visualise the estimated word importance for classifications.

As a result, words of texts are highlighted in green or in red. Green markings indicate word attributions for a positive classification (i.e. LAT) and red markings indicate word attributions for a negative classification (i.e. non-threat). Using this qualitative approach, texts can be analysed to discern which words contribute towards a particular classification.

Chapter 5

Results

In this chapter, the results of the experiments detailed in Chapter 4 are reported. These experiments consist of training four benchmark models alongside a BERT-based language model. Section 5.1 discusses the findings regarding the IRR. Section 5.2 takes a closer look at the baseline models' and language model's performance in terms of their F1-scores and PR-AUC. Section 5.3 reviews the performance of a language model trained on internal police data. Section 5.4 discusses the findings regarding the correct and incorrect classifications of the language model.

5.1 Inter-rater Reliability

To verify the adequacy of the annotation guidelines, an expert was consulted in the NP. Together with this expert, the annotation guidelines were created by discussing the practical application of the Dutch penal code regarding threats. Afterwards, a second annotator was found that was willing to aid with applying the annotation guide on the collected datasets. The second annotator is a master's student in law.

The annotation was performed on the preprocessed data, thus eliminating

names, Twitter handles, emojis, hashtags, and other identifying means or noise from the text.

5.1.1 Statistical significance of sample size

Prior to undertaking the process of annotation and calculation of Cohen's Kappa, a calculation was performed to determine statistical power. This is to state a probability of detecting a statistically significant kappa value and for providing a significant confidence interval for the kappa value. Initially, a value of 366 was calculated to have some statistical significance. However, this value did not consider the proportion of positive-to-negative values. To ameliorate this, the work of Sim & Wright is used (2005). They propose the use of a table where the minimum number of samples are given to detect a statistically significant kappa value. This table is based on the proportion of positive-to-negative ratings, the desired minimum Kappa value to detect (i.e. that is based on the context in which the study is performed), and the desired statistical power of the number of samples.

Since there is no ground truth available for the annotation of LATs, the mean is taken from the proportion of positive-to-negative ratings of both raters per dataset. Equation 5.1 provides the answer for the Dutch threat dataset and 5.2 provides the answer for the English threat dataset.

$$positive - to - negative_{NL} = \frac{\frac{(561)}{992} + \frac{(544)}{992}}{2} = 0.557 \tag{5.1}$$

$$positive - to - negative_{EN} = \frac{\frac{(388)}{2030} + \frac{(30)}{n}}{186} = 0.176 \tag{5.2}$$

For the Dutch dataset, the closest value to 0.557 in the table presented by Sim & Wright (2005) is 0.50. Therefore, this value was chosen. The desired minimum Kappa value to detect was chosen to be 0.80 as this was the specified value the NP

minimally required. This is due to the importance of the classifications and the possible detrimental consequences of wrongful classifications. Lastly, the null value for the two-tailed test was set at 0.70 with the amount of samples at 90% power. This null value was chosen to prevent type I errors as this makes it harder to reject the null hypothesis by mistake. The statistical power of 90% was chosen to minimize type II errors as the likelihood of detecting a true effect, if one is present, increases when testing for a higher statistical power. The resulting minimum number of samples to evaluate to have a statistically significant kappa value is 536.

For the English dataset, the closest value to 0.174 in the table presented by Sim & Wright (2005) is 0.10. Therefore, this value was chosen. As stated, the desired minimum Kappa value to detect was chosen to be 0.80 as this was the specified value the NP minimally required. Additionally, the null value for the two-tailed test was set at 0.40 with the amount of samples at 80% power. The resulting minimum number of samples to evaluate to have a statistically significant kappa value is 180. These values are chosen to be lower as it is expected that the calculated IRR in terms of Cohen's Kappa will be lower than the Dutch dataset's Cohen's Kappa. This is due to the higher class imbalance.

As was stated, the second annotator had to annotate roughly 536 texts for the Dutch dataset and 180 texts for the English dataset. With these annotations, it is possible to calculate Cohen's Kappa with a reasonable confidence interval for both datasets. However, the second annotator was able to annotate more texts than initially planned for the Dutch dataset, namely 992. For the English dataset, the second annotator annotated 186 texts.

For the 992 doubly annotated Dutch texts, a confusion matrix is presented in Figure 4. For the 186 doubly annotated English texts, a confusion matrix is presented in Figure 5. These confusion matrices present the number of equivalent

and opposite annotations between the two raters for the respective dataset.

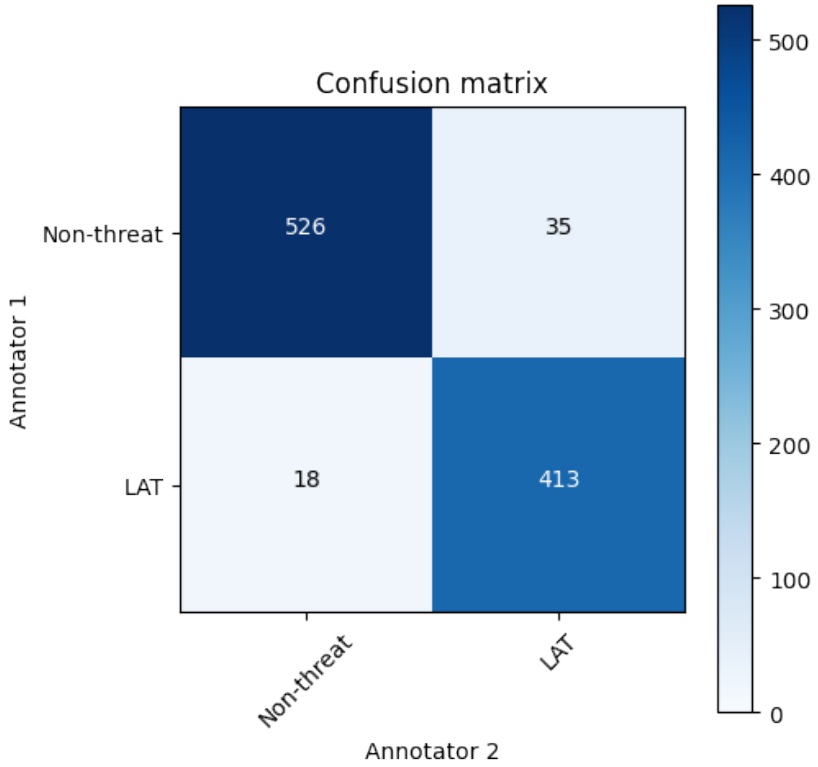

Figure 4: Confusion matrix for the annotation Dutch threats

5.1.2 Calculating Cohen's Kappa

In addition to calculating Cohen's Kappa for the IRR, its confidence interval and standard error are estimated. For these calculations, the following formulae are used, which are Equations 5.3, 5.4, and 5.5 respectively. These equations are found in the work of Sim & Wright (2005).

$$\kappa = (p_o - p_e)/(1 - p_e) \tag{5.3}$$

f

$$CI_\kappa = \kappa \pm z_{\alpha/2} SD(\kappa) \tag{5.4}$$

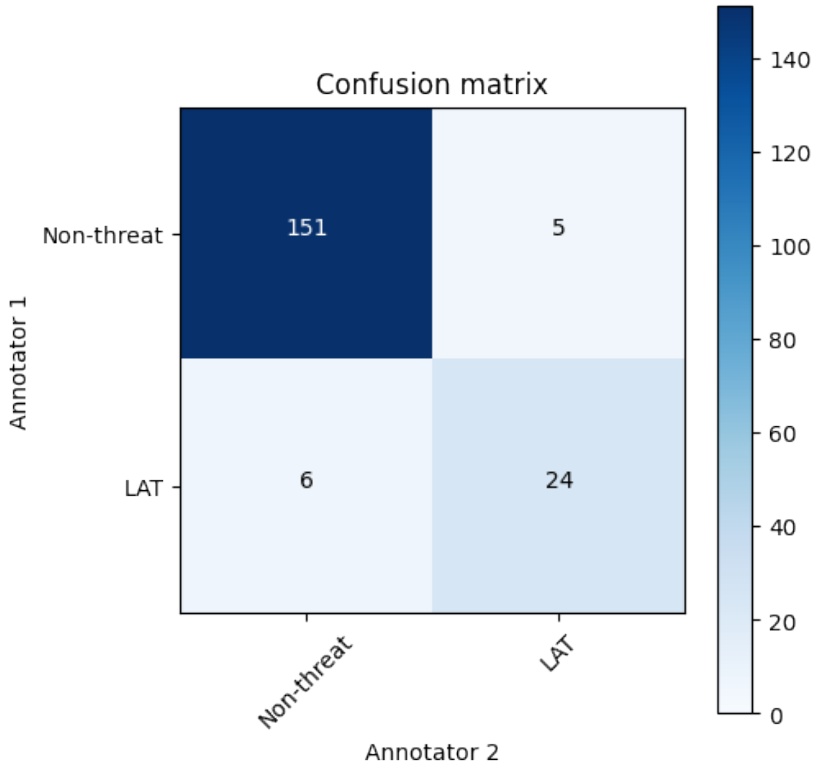

Figure 5: Confusion matrix for the annotation English threats

$$SE_\kappa = \sqrt{\frac{p_0(1-p_o)}{N(1-p_e)^2}} \tag{5.5}$$

Where p_o is the relative observed agreement among the raters and p_e is the hypothetical probability of chance agreement [77]. Additionally, the confidence interval is based on a two-sided 95% confidence interval.

For the Dutch dataset, the resulting Cohen's Kappa score was 0.890 with a confidence interval of [0.863, 0.920] and a standard error of 0.014.

For the English dataset, the resulting Cohen's Kappa score was 0.775 with a confidence interval of [0.647, 0.903] and a standard error of 0.065.

5.1.3 Determining the sufficiency of Cohen's Kappa

To determine the sufficiency of the IRR in terms of Cohen's Kappa, several benchmarks have been provided in the literature [13; 14; 15; 16]. However, these benchmarks are deemed arbitrary by Sim & Wright (2005) due to their negation of prevalence and bias. Prevalence indicates the degree to which a sign is present in the data. When raters encounter a binary annotation task (i.e. non-threats or LAT), a prevalence effect can exist when the proportion of agreements between the raters on the positive classification differs from that of the negative classification. The prevalence index ranges from [0, 1]. Therefore, if the prevalence index is close to 1, Cohen's Kappa is lower than when the prevalence index is close to 0. The effect of prevalence on Cohen's Kappa is greater for large values of Kappa than for small values [77]. Conversely, bias indicates the extent to which the raters disagree on the proportion of positive or negative cases. When there is a large bias, kappa is higher than when bias is low or absent. Therefore, it is undesirable to have high bias. The bias index also ranges from [0, 1]. However, kappa is higher when the bias index is close to 1, than when the bias index is close to 0. This results in the fact that the effect of bias is greater when kappa is small than when it is large.

To address prevalence and bias in this study, the prevalence index and bias index are computed and interpreted to give an assessment of their influence on Cohen's Kappa. To compute the prevalence index (pi) and bias index (bi), the Equations 5.6 and 5.7 are used: [77]

$$pi = \frac{(a - d)}{n} \tag{5.6}$$

$$bi = \frac{(b - c)}{n} \tag{5.7}$$

Where *a, b, c,* and *d* refer to the cells within the confusion matrix as presented

in Figures 4 and 5 They are specified from top left (i.e. a) to bottom right (i.e. d).

The resulting prevalence index and bias index for the Dutch dataset are:

$$pi = \frac{(526 - 413)}{992} = 0.100 \tag{5.8}$$

$$bi = \frac{(35 - 18)}{992} = 0.002 \tag{5.9}$$

The resulting prevalence index and bias index for the English dataset are:

$$pi = \frac{(151 - 24)}{186} = 0.683 \tag{5.10}$$

$$bi = \frac{(5 - 6}{186} = -0.005 \tag{5.11}$$

Sim & Wright (2005) argue that when the prevalence and/or bias index are high (i.e. a value close to either 0 or 1), a prevalence-adjusted bias-adjusted kappa (PABAK) should be reported in conjunction with the original kappa value. This is to inform the reader of the potential influence of high prevalence and/or bias index that is present in the data. Since the prevalence index is high (i.e. close to 0) in the Dutch dataset, the a and d cells are averaged and the kappa is calculated again. Therefore, the calculated PABAK for the Dutch dataset with the adjusted a and d cells is 0.896. This process is identical for the English dataset as the prevalence is also high. The calculated PABAK for the English dataset is 0.762.

Since no consensus exists for appropriate (Cohen's) Kappa scores, it must be tested against a value that represents a minimum acceptable level of agreement [77]. The minimum acceptable level of agreement indicated by the NP is 0.8. Since 0.8 lies lower than the estimated confidence interval of the Dutch dataset(i.e.

[0.863, 0.920]), it can be said that the achieved Cohen's Kappa is sufficient for this dataset and task. Additionally, for the English dataset, 0.8 lies between the estimated confidence interval (i.e. [0.647, 0.903]. Therefore, it can be said that the achieved Cohen's Kappa is insufficient for this dataset and task.

The final result of the annotation task for the Dutch dataset is that of the initial 5098 alleged death threat tweets as described in Section 3.1.1, 2325 texts were annotated as an LAT. The other 2773 texts were annotated as non-threats and were kept in the corpus. The final result of the annotation task for the English dataset is that of the initial 2030 alleged threats as described in Section 3.3.1, 388 texts were annotated as an LAT. The other 1642 texts were annotated as non-threats and were kept in the corpus.

5.2 Model performance: Baseline models vs. Language model

During the initial training phase of the models, it was noted that training the models with a class imbalance of 1/100 ensured that all models were overfitting on the training data. All models achieved a weighted F1-score of 0.5. Meaning, all models were overfitting for the majority class and were unable to detect any positive samples. To counteract this, a loss function was specified that made the model treat both the minority and majority class equally when possible (i.e. binary cross-entropy). Despite that effort, the models were still overfitting. Therefore, the class imbalance used to train the models was eventually reduced to roughly 1/20 as this prevented overfitting by all models. This means that the 2325 annotated LATs were supplemented with 42500 randomly scraped Dutch tweets.

After training and the concurrent hyperparameter tuning of the models with the stratified k-fold approach as specified in Section 4.2, the final set of hyperpa-

rameters per model were discovered. Appendix C presents these results.

5.2.1 Analysing performance metrics

After training the baseline models and the BERT-based language model on the constructed datasets, the models were evaluated. However, it was found that some texts were annotated incorrectly during the inspection of the results. By means of post-correction, this was ameliorated by consulting the annotation guide and re-annotating these texts and evaluating the models again. There were eleven texts that were annotated wrongly for the Dutch dataset and seven texts for the English dataset.

The results consisted of a collection of macro & micro F1-scores, F1-scores per class, and PR-AUC scores per classifier. For the results on the Dutch dataset, the means of these values are shown in Figures 6, 7, 8, and 9 respectively. The performance metrics per fold, the PR-AUC graph most representative of the classifiers and a boxplot illustrating the skewness and variance of the PR-AUC scores per classifiers are presented in D. For the results on the English dataset, the means of these values are shown in Figures 10, 11, 12, and 13 respectively. The performance metrics per fold, the PR-AUC graph most representative of the classifiers and a boxplot illustrating the skewness and variance of the PR-AUC scores per classifiers are presented in E.

5.2.1.1 Statistical tests on results of Dutch dataset

Tukey's HSD Test for multiple comparisons found that the mean value of the macro F1-score was significantly different between the BERT model and the BiLSTM model (p = 0.00, 95% C.I. = [-0.25, -0.22]), Naive Bayes model (p = 0.00, 95% C.I. = [-0.03, 0.00]), and SVM model (p = 0.00, 95% C.I. = [0.01, 0.04]). There was no statistically significant difference between the BERT model and the CNN

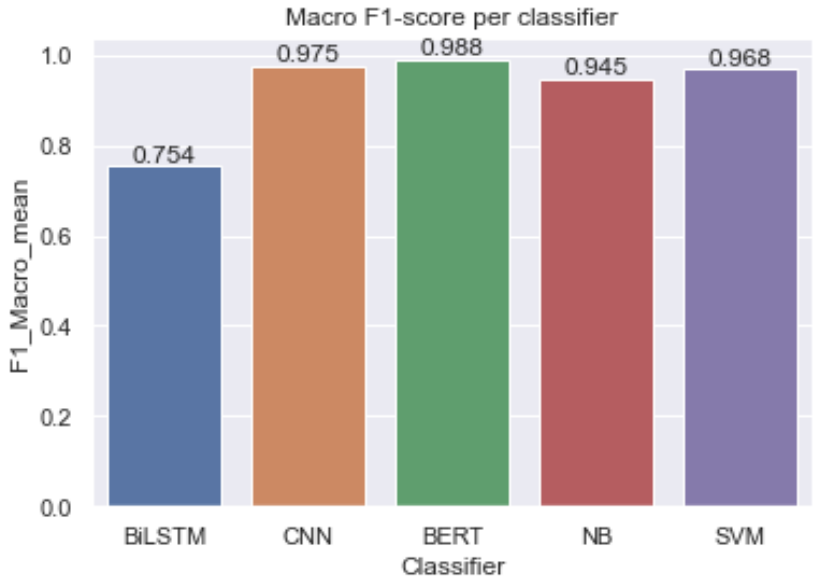

Figure 6: Mean Macro F1-scores per classifier for the Dutch dataset

model (p=0.11).

Tukey's HSD Test for multiple comparisons found that the mean value of the micro F1-score was significantly different between the BERT model and the BiLSTM model (p = 0.00, 95% C.I. = [-0.05, -0.05]), Naive Bayes model (p = 0.00, 95% C.I. = [-0.01, 0.00]), and SVM model (p = 0.02, 95% C.I. = [-0.01, 0.00]). There was no statistically significant difference between the BERT model and the CNN model (p=0.06).

Tukey's HSD Test for multiple comparisons found that the mean value of the majority class F1-score was significantly different between the BERT model and the BiLSTM model (p = 0.00, 95% C.I. = [-0.03, -0.03]), Naive Bayes model (p = 0.00, 95% C.I. = [-0.01, 0.00]), and SVM model (p = 0.02, 95% C.I. = [0.00, 0.00]). There was no statistically significant difference between the BERT model and the CNN model (p=0.09).

Tukey's HSD Test for multiple comparisons found that the mean value of the minority class F1-score was significantly different between the BERT model and

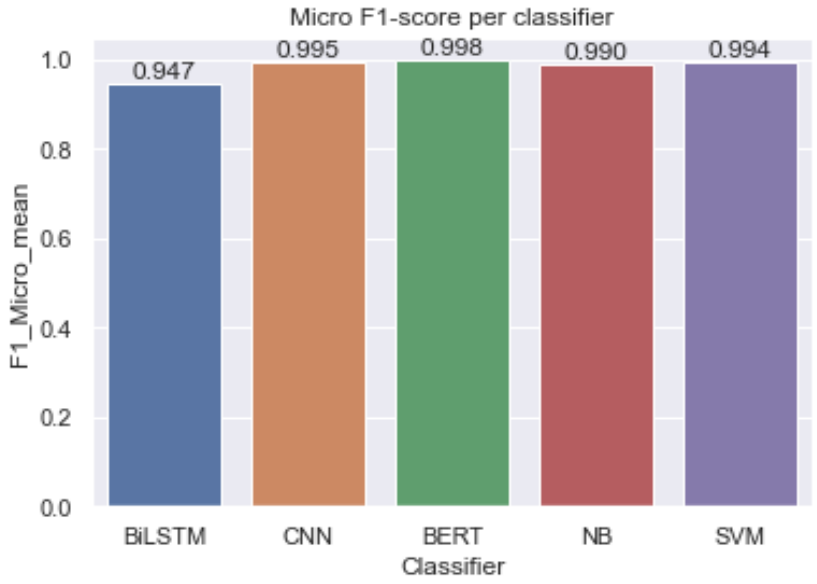

Figure 7: Mean Micro F1-scores per classifier for the Dutch dataset

the BiLSTM model (p = 0.00, 95% C.I. = [-0.47, -0.41]), Naive Bayes model (p = 0.00, 95% C.I. = [-0.11, -0.05]), and SVM model (p = 0.02, 95% C.I. = [-0.07, -0.01]). There was no statistically significant difference between the BERT model and the CNN model (p=0.10).

Tukey's HSD Test for multiple comparisons found that the mean value of the PR-AUC score was significantly different between the BERT model and the BiLSTM model (p = 0.00, 95% C.I. = [-0.46, -0.40]), Naive Bayes model (p = 0.00, 95% C.I. = [-0.10, -0.05]), and SVM model (p = 0.02, 95% C.I. = [-0.07, -0.01]). There was no statistically significant difference between the BERT model and the CNN model (p=0.12).

5.2.1.2 Statistical tests on results of English dataset

Tukey's HSD Test for multiple comparisons found that the mean value of the macro F1-score was significantly different between the BERT model and the BiLSTM model (p = 0.01, 95% C.I. = [-0.25, -0.03]), CNN model (p = 0.02, 95% C.I. =

[-0.23, -0.01]), Naive Bayes model (p = 0.00, 95% C.I. = [-0.36, -.0.14]), and SVM model (p = 0.00, 95% C.I. = [-0.26, -0.04]).

Tukey's HSD Test for multiple comparisons found that the mean value of the micro F1-score was not significantly different between the BERT model and the BiLSTM model (p = 0.78, 95% C.I. = [-0.04, 0.02]), CNN model (p = 0.62, 95% C.I. = [-0.05, 0.02]), Naive Bayes model (p = 0.38, 95% C.I. = [-0.05, 0.01]), and SVM model (p = 0.37, 95% C.I. = [-0.05, 0.01]).

Tukey's HSD Test for multiple comparisons found that the mean value of the majority class F1-score was not significantly different between the BERT model and the BiLSTM model (p = 0.73, 95% C.I. = [-0.02, 0.01]), CNN model (p = 0.64, 95% C.I. = [-0.02, 0.01]), Naive Bayes model (p = 0.44, 95% C.I. = [-0.03, 0.01]), and SVM model (p = 0.36, 95% C.I. = [-0.03, 0.01]).

Tukey's HSD Test for multiple comparisons found that the mean value of the minority class F1-score was significantly different between the BERT model and the BiLSTM model (p = 0.05, 95% C.I. = [-0.40, -0.00]), Naive Bayes model (p = 0.00, 95% C.I. = [-0.63, -0.23]), and SVM model (p = 0.02, 95% C.I. = [-0.42, -0.02]). There was no statistically significant difference between the BERT model and the CNN model (p=0.99).

Tukey's HSD Test for multiple comparisons found that the mean value of the PR-AUC score was significantly different between the BERT model and the Naive Bayes model (p = 0.00, 95% C.I. = [-0.47, -0.11]), and the SVM model (p = 0.02, 95% C.I. = [-0.38, -0.02]). There was no statistically significant difference between the BERT model and the BiLSTM model (p=0.14) and the CNN model (p=0.14).

5.3 Model performance: Language model on internal police data

The supervisor at the NP applied the annotation guide as specified in Appendix B to annotate this dataset. Afterwards, the baseline models and the BERT model were used to predict the classes for the texts of the dataset.

5.3.1 Analyzing performance metrics

After training the baseline models and the BERT-based language model on the constructed Dutch dataset, the models were evaluated on the internal police data. It was unfortunately impossible to train the models on the internal police data. The reason for this was the fact that the data were too sensitive and thus the NP were not willing.

The results in Table 4. The PR-AUC graphs per classifier are presented in Appendix F.

Metric	**BiLSTM**	**CNN**	**LM**	**NB**	**SVM**
F1 Macro	0.498	0.492	**0.582**	0.500	0.500
F1 Micro	**0.999**	0.969	0.995	**0.999**	**0.999**
F1 Majority	**0.999**	0.984	0.997	**0.999**	**0.999**
F1 Minority	0.000	0.000	**0.167**	0.000	0.000
PR-AUC	**0.500**	0.000	0.300	0.000	**0.500**

Table 4: Performance metrics on internal police data

It was impossible to provide a meaningful statistical test for these performance metrics as there were only two LATs in the whole dataset of the NP.

5.4 Understanding language model classifications

To understand the classifications made by the language model, the classifications are interpreted with the help of a feature relevance estimation approach. This approach is taken for the results on the Dutch threat corpus and the internal police data. For each dataset, the top 10 worst classifications and top 10 best classifications are used for input.

Using the feature relevance estimation module from the Captum framework for interpreting deep learning models, it was possible to visualise the estimated word importance for classifications.

As a result, words are highlighted in green or in red. Green markings indicate word attributions for a positive classification (i.e. LAT) and red markings indicate word attributions for a negative classification (i.e. non-threat). Furthermore, as a result of the language model's architecture, a token delimiter (i.e. Ğ) is added which separates tokens. However, when examples are given, the original text will also be present for readability. Appendices G and G respectively present ten examples of texts and their feature relevance estimations where the language model correctly or incorrectly classified the target class with high probability for the Dutch dataset.

5.4.1 Observations on the Dutch dataset

5.4.1.1 Word importance for LAT classifications

Due to the homogeneity of threats in the Dutch dataset, most LATs involve a threat to murder someone. Therefore, the language model has learned to recognize the threat when it comes in the form of:

- Ik ga jou + *verb indicating a threat, e.g. vermoorden* | I will + *verb indicating a threat, e.g. kill* you

- Jij gaat + *word indicating a threatening state, e.g. dood* | You will + *word indicating a threatening state, e.g. die*

These are two examples of threats that occur most often in the constructed Dutch dataset. Other types of threats that are present involved arson to a person or a person's property, indecent assault, aggravated assault, or kidnapping.

Therefore, a sentenced structured with a pronoun at the beginning, more specifically *I/Ik*, gets an initial high word attribution score when combined with verb that indicates a threat. This is independent of the remaining contents of the sentence. Additionally, another pronoun, *You/Jij* combined with the verb *are going to/gaat* also yields high word attribution scores regardless of the remaining contents of the sentence.

Aside from these two possible structures of sentences, there are individual words that when present in a sentence score very high for the overall word attribution scores. Examples of these words (and their possible permutations) are *neersteken/to stab*, *brand/fire*, *fik/fire*, *bom/bomb*, *school/school*, *vermoorden/to kill*, *haat/hate*, *schieten/to shoot*, *dood/death*, *kapot/broken*, *blazento blow away (slang)*, and *slopen/to demolish.*

Another observation is that whenever a word is present that could be highly threatening, such as *steken/to stab* and *schieten/to shoot*, it receives a high positive word attribution score. However, these words by themselves do not signify an LAT.

5.4.1.2 Word importance for non-threat classifications

For non-threat classifications, it was observed that when texts do not follow the one of the two structures detailed in the previous paragraph, they are more likely to be classified as a non-threat. When that is the case, there are more uncommon words in these texts. Examples of these words are: *zometeen/in a moment*, *even/for a*

bit, *vrees/fear*, *gwn (slang)/normal*, *nee/no*, and *nog/still*, and any word combined with *kanker/cancer*, any word combined with *kut/pussy*.

5.4.2 Observations on the internal police data

The same approach was used for understanding the classifications of the language model on the internal police data. Albeit that it is impossible to reproduce the texts here due to confidentiality constraints, I have observed the results of the XAI approach and I will detail my observations below.

5.4.2.1 Word importance for non-threat classifications

For non-threat classifications, a wide range of words and their respective placing a sentence determined their negative word attribution. Below is a summary of words and their possible placing in a sentence that result in a negative word attribution.

- *Heb jij / Have you*
- *Jij / You* (When not placed at the beginning of a sentence)
- *Trouwens / By the way*
- *Steeds / Still*
- *Broer / Brother*
- *Vriend / Friend* (Inconsistent word attribution, no clear sign of when it matters)
- *Morgen / Tomorrow* (Inconsistent word attribution, no clear sign of when it matters)

5.4.2.2 Word importance for LAT classifications

It was observed that when the same two structures are recognized in the text that are detailed in the previous section, the text is more likely to be classified as an LAT.

Aside from these two possible structures of sentences, there are individual words that when present in a sentence score very high for the overall word attribution scores. Examples of these words (and their possible permutations) are *verkrachten/to rape*, *kogel/bullet*, *kop,head*, *morgen/tomorrow* (only when this is placed after a verb), *dood*, *death* (when combined with a verb), *ak*, *aks*, *wapens/weapons*, *ik sta bij/I am standing at*, *kom naar/come to*, *uithaal/flick*.

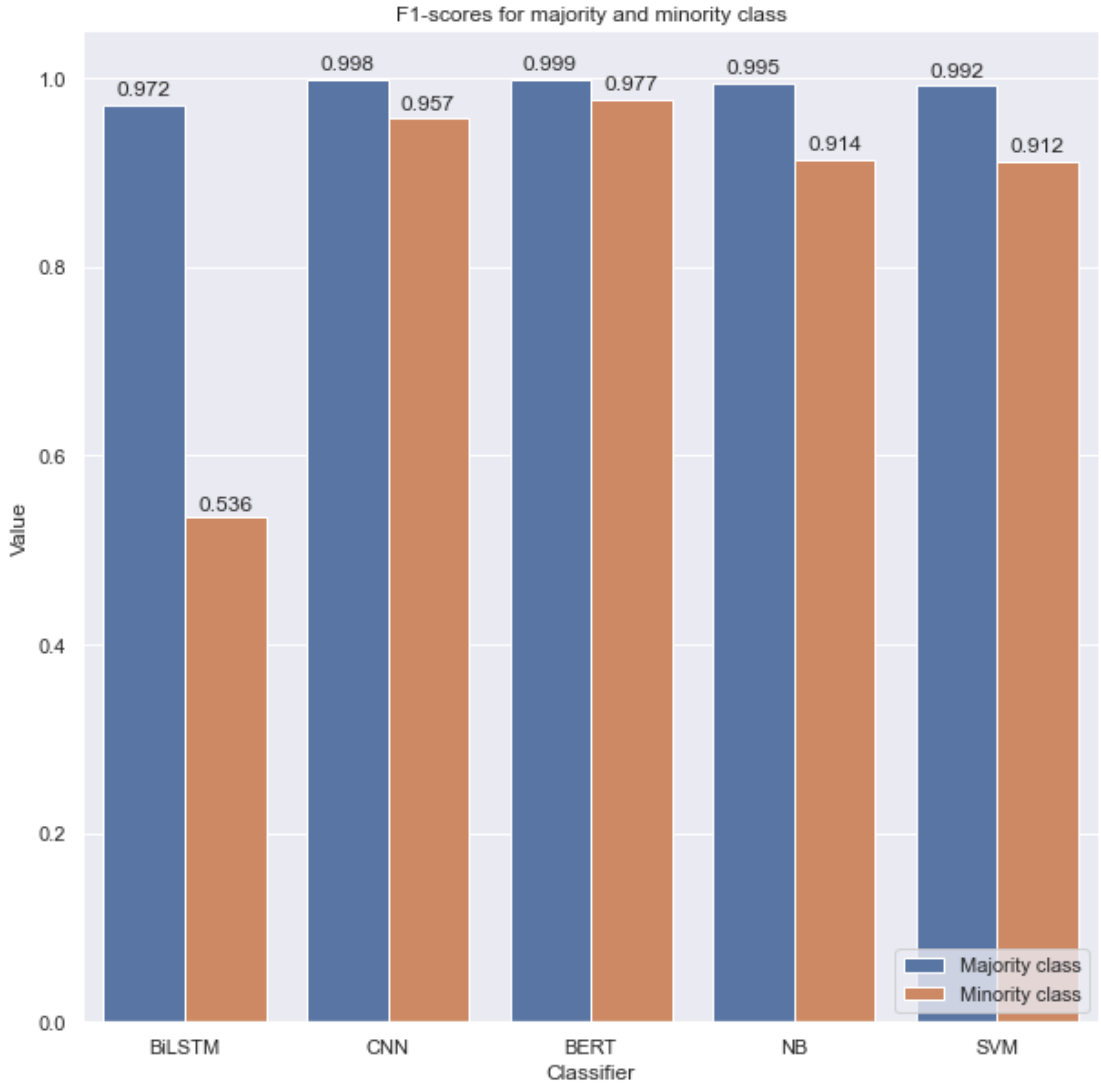

Figure 8: Mean F1-scores per class per classifier for the Dutch dataset

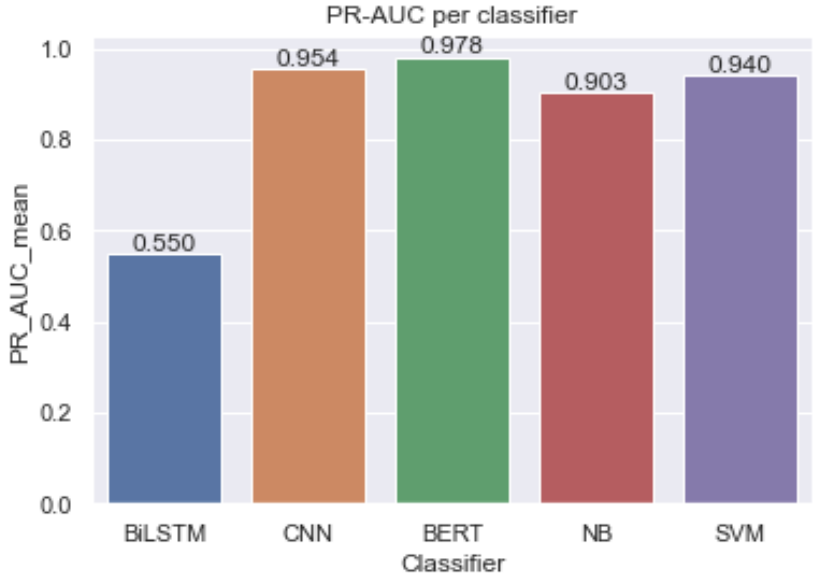

Figure 9: Mean PR-AUC scores per classifier for the Dutch dataset

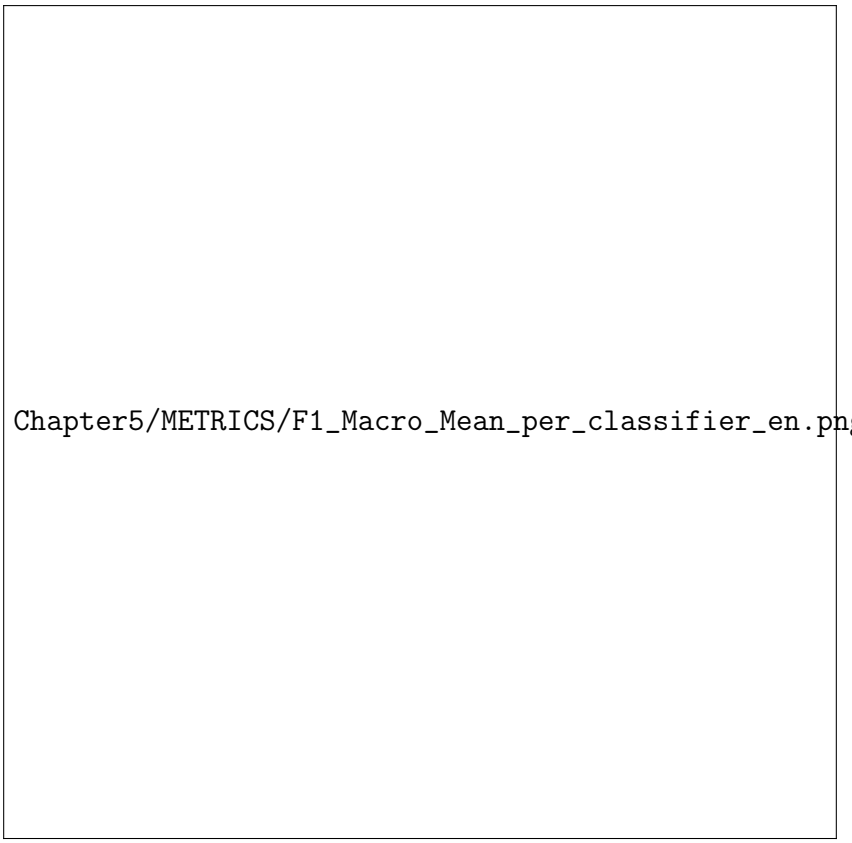

Figure 10: Mean Macro F1-scores per classifier for the English dataset

Figure 11: Mean Micro F1-scores per classifier for the English dataset

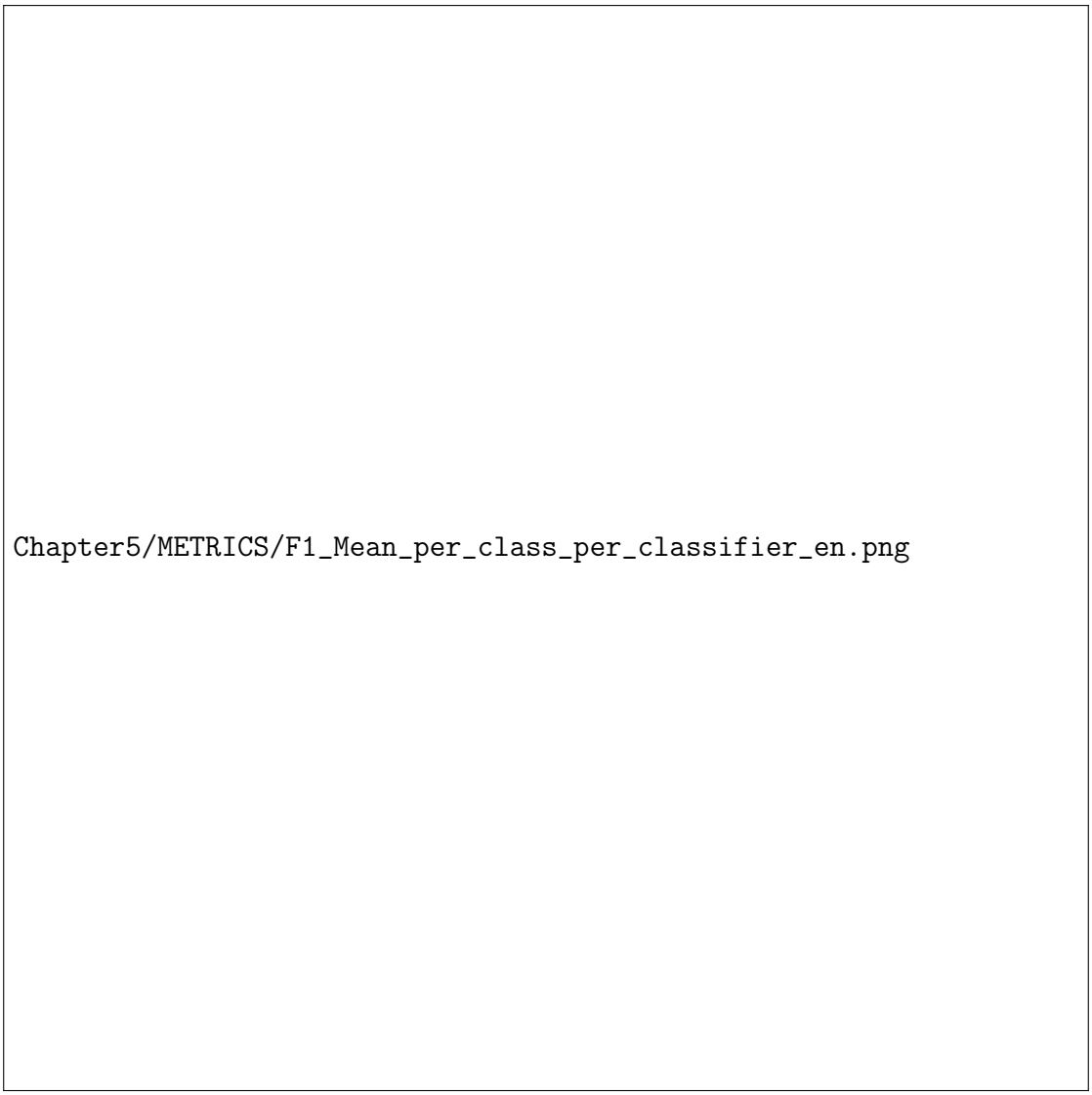

Figure 12: Mean F1-scores per class per classifier for the English dataset

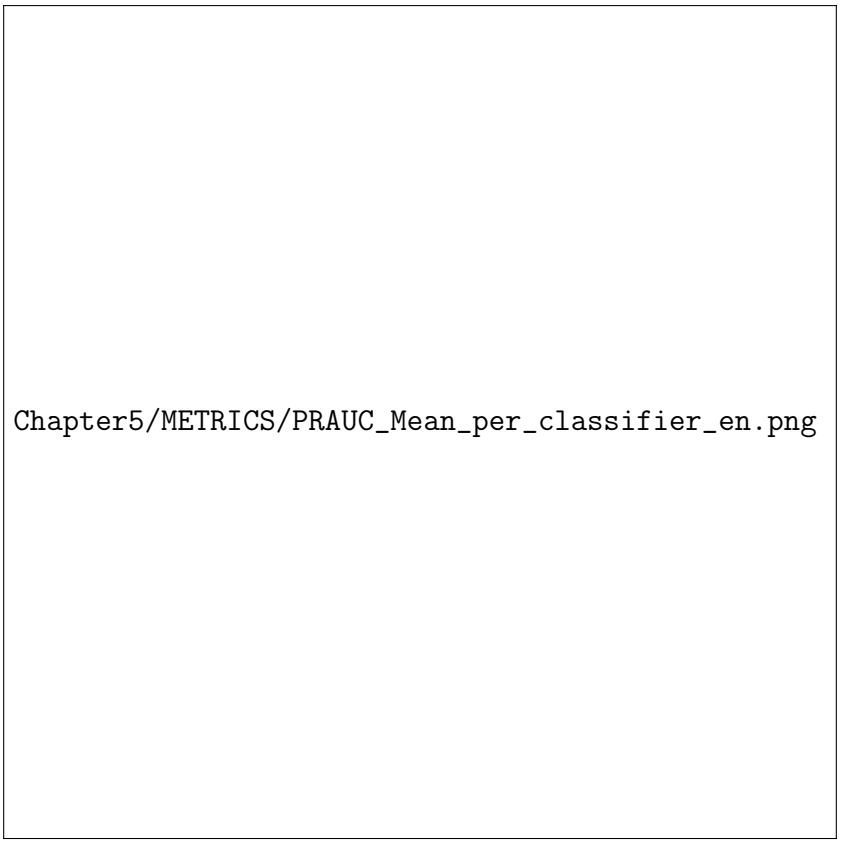

Figure 13: Mean PR-AUC scores per classifier for the English dataset

Chapter 6

Discussion

In this chapter, the main research question will be answered by formulating answers to the sub-questions. Afterwards, the limitations of this thesis are acknowledged. Based on these limitations and the results of the thesis, suggestions for future work are made.

6.1 Conclusion

The main question in this thesis was to what extent the Dutch legal definition of a threat can be learned algorithmically by language models. This research question was divided into four sub-questions which will first be answered and discussed. Afterwards, an answer to the main research question will be formulated and discussed.

RQ$_1$: Is the inter-rater reliability (IRR) in terms of Cohen's Kappa caused by chance agreement and is it sufficient for the annotation of LATs?

There was enough evidence to conclude that the IRR in terms of Cohen's Kappa was not caused by chance for both the Dutch dataset and the English dataset. For the Dutch dataset, the second annotator was able to annotate more texts than

required to calculate the Cohen's Kappa with sufficient statistical power. This resulted in a Cohen's Kappa score of 0.890 (PABAK: 0.896) with a confidence interval of [0.863, 0.920] and a standard error of 0.014. For the English dataset, the Cohen's Kappa score was calculated at 0.775 (PABAK: 0.762) with a confidence interval of [0.647, 0.903] and a standard error of 0.065.

To assess the sufficiency of the Cohen's Kappa score, a small literature study was performed. It can be concluded that no consensus exists for an adequate Kappa score. Sim and Wright argue that the calculated Kappa score must be tested against a value that represents a minimum acceptable level of agreement within the problem domain [77]. The minimum acceptable level of agreement indicated by the NP is 0.8. Since 0.8 lies lower than the estimated confidence interval (i.e. [0.863, 0.920]), it can be said that the achieved Cohen's Kappa for the Dutch dataset is sufficient for this task. However, since the minimum acceptable level lies within the confidence interval of Cohen's Kappa for the English dataset (i.e. 0.762), it is insufficient for use at the NP.

RQ_2: Do the language model and benchmark models differ statistically significantly in terms of their F1-score and PR-AUC score with respect to the classification of LATs?

For the Dutch dataset, there was enough evidence to conclude that the language model's performance in terms of its F1-scores and PR-AUC score differs statistically significantly with the benchmark models, aside from the CNN model in all instances. A Tukey HSD test for multiple comparison was performed to compare the mean values of all folds per performance metric. For each performance metric, the language model outperformed the BiLSTM, Naive Bayes, and SVM classifiers. Albeit that the language model achieved higher scores than the CNN classifier in all performance metrics, these differences were not statistically significant.

For the English dataset, there was enough evidence to conclude that the language model's performance in terms of its F1-scores and PR-AUC score differs statistically significantly with the benchmark models, aside from the CNN model and BiLSTM model in a minimal number of instances. A Tukey HSD test for multiple comparison was performed to compare the mean values of all folds per performance metric. For each performance metric, the language model outperformed the BiLSTM, Naive Bayes, and SVM classifiers. Albeit that the language model achieved higher scores than the CNN classifier in all performance metrics, these differences were not statistically significant.

RQ$_3$: Do the language model and benchmark models differ statistically significantly in terms of their F1-score and PR-AUC score with respect to the classification of LATs in internal police data?

There were not enough data to conduct a statistical test on the significance of results on the internal police data; only two legally actionable threats were present in that dataset. However, a conclusion can be drawn that in the limited testing pool of the internal police data, the language model performed better in terms of correctly classifying the minority class. This is signified by a higher F1-Macro score and F1-Minority score. On all other performance metrics, it was outscored by the Naive Bayes, SVM, and BiLSTM classifiers. In this testing instance, the CNN does not appear to achieve similar results as the language model.

RQ$_4$: When does the language model correctly and incorrectly classify threats and non-threats?

On the Dutch dataset, it was observed that the language model has little problem correctly classifying death threats. This is especially the case when they come in a frequent structure of a sentence such as: *Ik ga je doodmaken/I will kill you*. This is to be expected as this structure was observed many times during the annotation process.

However, when a threat is more in line with aggravated assault, indecent assault, kidnapping, or a terrorist threat, the language model finds it harder to correctly classify the text. This is exemplified by Figures 42, 44, 45, 46, 48, and 50 in Appendix **??**.

Furthermore, it sometimes occurred albeit that the text is deemed an LAT, the text was classified incorrectly despite having a number of highly positive word attributions. An example of this is Figure 46 in Appendix **??**. Despite the fact that this text features the phrase: *"KOM WE MAKEN DIE KUTKOP DOOD NA SCHOOL"*, this is offset by negative word attributions in the same sentence such as: *HAHAHAH*, *MENNO, IN*, *DE*, and *MAKEN*. This means that is possible for sentence that contain an LAT to be classified as a non-threat if enough negative word attributions are present in the same text.

MRQ: To what extent can the Dutch legal definition of a threat be learned algorithmically by language models? For the Dutch dataset, there is enough evidence to suggest that it is possible to encapsulate the Dutch legal definition of a threat in annotation guidelines to construct a corpus of legally actionable threats. With an achieved Cohen's Kappa score of 0.890 (PABAK: 0.896, see Section 5.1.3), a CI of [0.863, 0.920], and a minimum accepted level of 0.8 at the target institution, it can be concluded that the Dutch legal definition can be operationalized within the context of Machine Learning for this dataset.

For the English dataset, there is not enough evidence to suggest that it is possible to encapsulate the Dutch legal definition of a threat in annotation guidelines to construct a corpus of legally actionable threats. With an achieved Cohen's Kappa score of 0.775 (PABAK: 0.762, see Section 5.1.3), a CI of [0.863, 0.920], and a minimum accepted level of 0.8 at the target institution, it can be concluded that the Dutch legal definition cannot be operationalized within the context of Machine Learning for this dataset.

This difference in results is most likely caused by the nature of these two datasets. This difference being that the Dutch dataset is comprised of mostly death threats, while the English is far more heterogeneous in its threats. However, it can also be concluded that this approach can yield reproducible results. Lastly, when more studies are done in this domain, this approach could allow researchers to generalise results.

After performing statistical tests on the performance metrics per classifier and per dataset, it can be concluded that the language model statistically significantly outperforms the benchmark models. For the Dutch dataset, only the CNN model was not outperformed on all performance metrics. For the English dataset, the language model outperformed all benchmark models on the F1-Macro score. And, aside from the CNN model and BiLSTM model, it performed all the benchmark models on the other performance metrics (i.e. F1-Micro, F1-Minority class, F1-Majority class, and PR-AUC score). The language model did not statistically significantly outperform the CNN model on any performance metric and it did not statistically significantly outperform the BiLSTM on the PR-AUC score.

Furthermore, the language model attributed high feature importance to the appropriate sentence structures for a threat. It performed particularly well on death threats, since this type of threat formed the majority of alleged threats in the Dutch threat corpus. The language model more often incorrectly classified threats when they were referencing to aggravated assault, indecent assault, kidnapping, or a terrorist threat. It does show that the words indicating this (e.g. permutations of *verkrachten/to rape*) receive a positive word attribution score, but this is offset or nullified by other words in the sentence thereby transforming the classification into a non-threat.

Finally, when applying this methodology to a real-world setting, namely the internal police data, there are only tentative results. These results being that

despite no models were trained on the data that were tested on, the language model was able to outperform the benchmark models in the F1 Macro score and F1 Minority class score (i.e. the most important metrics according to the NP). The language model did not outperform the benchmark models in terms of the F1 Micro score, F1 Minority class score, and PR-AUC score.

6.2 Limitations

The first limitation of this research is the training phase of the models. Albeit that the NP intended for a class imbalance of 1/100, the resulting class imbalance was 1/20. Although this prevented the models from overfitting on the training data, it does not accurately reflect the real-world. Additionally, for some unknown reason, it was impossible to train an accurate BiLSTM classifier. Therefore, this classifier should be seen as an informed guesser as it did overfit on the majority class while occasionally predicting the correct class for the minority class.

The second limitation was the internal police data. The main problem was that I did not have direct access to the data. This significantly slowed down the thesis as all access had to occur indirectly via my supervisor at the National Police. On top of this, it was only possible to extract performance measures and console outputs from the scripts I supplied my supervisor with, because of the confidentiality constraints of the data. This prevented me from fully understanding the data and catching errors at the same time as they occurred. Speaking of these errors, the data were not pre-processed properly. This is easily seen when investigating Figure 2 and Table 2. As said, the texts of the internal police data are best described as telecommunications between humans that resemble e-mails. These messages therefore have headers so that the service handler knows where to send the message to. However, not all headers were properly cleaned out of the

texts resulting in skewed data. Additionally, the number of LATs in the dataset is minimal with a number of two LATs. It was expected to have a minimum of 20 to 30 LATs to test on, but this was not achieved. Since there were only two LATs in the whole dataset, it was impossible to perform a meaningful statistical test. Therefore, it was not feasible to provide any statistical test on these data.

The third limitation of this research was the inability to train the language model and the benchmark models on the internal police data. Due to confidentiality constraints for myself and time constraints for my supervisor at the NP, it was not feasible to get access to the data or, for my supervisor, to have time to properly clean the data. This further diminishes the validity of these results.

6.3 Future work

Despite the fact that many limitations have been mentioned, there are more things to acknowledge to guide future work in this direction. The primary one being the ethical side of applying ML techniques to solve judicial problems. It is one thing to create an algorithm and test its performance, but its another thing entirely to test its adequacy in a real-world setting. For example, it must be tested to see how it deals with jokes, sarcasm, and unbelievable threats. It must also be tested to see how the language model deals with input permutations. For example, will the language model be quicker to classify a text as an LAT when it mentions a man or a woman? Or, on a lower level, when you compare the two sentences: "When I see you tomorrow I will punch you so hard!" vs. "When I see you tomorrow I will hug you so hard!".

Additionally, the foundation of this research has been laid on the annotation guidelines and their adequacy. Albeit that these guidelines were assessed by my supervisor at the National Police, they must be evaluated by criminal lawyers to

gauge their adequacy. Since law is not an exact science, significant efforts must be made to ensure that the annotation guidelines are robust. This can be done by testing these annotation guidelines against edge cases (e.g. threats that judges do not agree on).

And lastly, an attempt should be made to train and test these models on real-world data which is assessed by a police entity.

References

[1] S. V. Georgakopoulos, S. K. Tasoulis, A. G. Vrahatis, and V. P. Plagianakos, "Convolutional neural networks for toxic comment classification," in Proceedings of the 10th hellenic conference on artificial intelligence, pp. 1–6, 2018. 1

[2] Z. Waseem, "Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter," in Proceedings of the first workshop on NLP and computational social science, pp. 138–142, 2016. 1

[3] G. K. Pitsilis, H. Ramampiaro, and H. Langseth, "Detecting offensive language in tweets using deep learning," arXiv preprint arXiv:1801.04433, 2018. 1

[4] C. Van Hee, G. Jacobs, C. Emmery, B. Desmet, E. Lefever, B. Verhoeven, G. De Pauw, W. Daelemans, and V. Hoste, "Automatic detection of cyberbullying in social media text," PloS one, vol. 13, no. 10, p. e0203794, 2018. 1

[5] M. Aldwairi and A. Alwahedi, "Detecting fake news in social media networks," Procedia Computer Science, vol. 141, pp. 215–222, 2018. 1

[6] P. Chakraborty and M. H. Seddiqui, "Threat and abusive language detection on social media in bengali language," in 2019 1st International Conference

on Advances in Science, Engineering and Robotics Technology (ICASERT), pp. 1–6, IEEE, 2019. 1, 2, 3, 25

[7] M. Spitters, P. T. Eendebak, D. T. Worm, and H. Bouma, "Threat detection in tweets with trigger patterns and contextual cues," in 2014 IEEE Joint Intelligence and Security Informatics Conference, pp. 216–219, IEEE, 2014. 1, 2, 3

[8] S. A. AlAjlan and A. K. J. Saudagar, "Machine learning approach for threat detection on social media posts containing arabic text," Evolutionary Intelligence, vol. 14, no. 2, pp. 811–822, 2021. 1, 2, 3, 25

[9] H. L. Hammer, "Detecting threats of violence in online discussions using bigrams of important words," in 2014 IEEE Joint Intelligence and Security Informatics Conference, pp. 319–319, IEEE, 2014. 2, 3, 24, 32

[10] C. E. Stenberg, "Threat detection in online discussion using convolutional neural networks," Master's thesis, 2017. 2, 3, 24

[11] J. Singh, B. McCann, N. S. Keskar, C. Xiong, and R. Socher, "Xlda: Cross-lingual data augmentation for natural language inference and question answering," arXiv preprint arXiv:1905.11471, 2019. 2

[12] T. Ranasinghe and M. Zampieri, "Multilingual offensive language identification for low-resource languages," Transactions on Asian and Low-Resource Language Information Processing, vol. 21, no. 1, pp. 1–13, 2021. 2

[13] J. L. Fleiss, B. Levin, and M. C. Paik, Statistical methods for rates and proportions. john wiley & sons, 2013. 4, 60

[14] D. G. Altman, Practical statistics for medical research. CRC press, 1990. 4, 60

[15] P. E. Shrout, "Measurement reliability and agreement in psychiatry," Statistical methods in medical research, vol. 7, no. 3, pp. 301–317, 1998. 4, 60

[16] J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," biometrics, pp. 159–174, 1977. 4, 60

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019. 5

[18] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," arXiv preprint arXiv:1911.02116, 2019. 5, 20

[19] M. Juuti, T. Gröndahl, A. Flanagan, and N. Asokan, "A little goes a long way: Improving toxic language classification despite data scarcity," arXiv preprint arXiv:2009.12344, 2020. 5

[20] H. Van, "Mitigating data scarcity for large language models," arXiv preprint arXiv:2302.01806, 2023. 5

[21] A. Silva, P. Tambwekar, and M. Gombolay, "Towards a comprehensive understanding and accurate evaluation of societal biases in pre-trained transformers," in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 2383–2389, 2021. 6

[22] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in Proceedings of the 18th international conference on evaluation and assessment in software engineering, pp. 1–10, 2014. 7, 8

[23] W. M. Bramer, G. B. De Jonge, M. L. Rethlefsen, F. Mast, and J. Kleijnen, "A systematic approach to searching: an efficient and complete method to develop literature searches," Journal of the Medical Library Association: JMLA, vol. 106, no. 4, p. 531, 2018. 7, 9

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," Advances in neural information processing systems, vol. 30, 2017. 8, 18, 19

[25] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning–based text classification: a comprehensive review," ACM Computing Surveys (CSUR), vol. 54, no. 3, pp. 1–40, 2021. 8, 14, 17

[26] T. Gales, "Threatening stances: a corpus analysis of realized vs. non-realized threats," Language and Law/Linguagem e Direito, vol. 2, no. 2, 2017. 8

[27] F. Poletto, V. Basile, M. Sanguinetti, C. Bosco, and V. Patti, "Resources and benchmark corpora for hate speech detection: a systematic review," Language Resources and Evaluation, vol. 55, no. 2, pp. 477–523, 2021. 8

[28] B. Fraser, "Threatening revisited," Forensic linguistics, vol. 5, pp. 159–173, 1998. 10, 11, 12, 26

[29] "Wetboek van strafrecht." 11, 105

[30] "Threat definition amp; meaning - black's law dictionary," Jan 2014. 11, 23

[31] "Wetboek van strafrecht." 11, 106

[32] "Wetboek van strafrecht." 11, 105

[33] J. R. Searle, Expression and meaning: Studies in the theory of speech acts. Cambridge University Press, 1979. 11

[34] P. Tiersma and L. M. Solan, "Speaking of crime: The language of criminal justice," 2005. 12

[35] J. P. Kaplan, "Case report: Elonis v. united states.," International Journal of Speech, Language & the Law, vol. 23, no. 2, 2016. 12

[36] A. Nini, "Register variation in malicious forensic texts," International Journal of Speech, Language and the Law, vol. 24, no. 1, 2017. 12

[37] D. Biber, "A typology of english texts," 1989. 12

[38] D. Biber, Variation across speech and writing. Cambridge University Press, 1991. 12

[39] T. A. Gales, Ideologies of violence: A corpus and discourse analytic approach to stance in threatening communications. University of California, Davis, 2010. 13

[40] E. D. Liddy, "Natural language processing," 2001. 13

[41] P. Hacker, R. Krestel, S. Grundmann, and F. Naumann, "Explainable ai under contract and tort law: legal incentives and technical challenges," Artificial Intelligence and Law, vol. 28, no. 4, pp. 415–439, 2020. 13

[42] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, et al., "Explainable

artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," Information fusion, vol. 58, pp. 82–115, 2020. 13, 21

[43] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," Information, vol. 10, no. 4, p. 150, 2019. 13, 14, 15

[44] C. Giovanelli, X. Liu, S. Sierla, V. Vyatkin, and R. Ichise, "Towards an aggregator that exploits big data to bid on frequency containment reserve market," in IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society, pp. 7514–7519, IEEE, 2017. 15

[45] D. S. Jasim, "Data mining approach and its application to dresses sales recommendation," Research Gate, 2015. 15

[46] H. Bansal, G. Shrivastava, G. N. Nguyen, and L.-M. Stanciu, Social network analytics for contemporary business organizations. IGI Global, 2018. 15

[47] C. Cortes and V. Vapnik, "Support vector machine," Machine learning, vol. 20, no. 3, pp. 273–297, 1995. 16

[48] T. Gonalves and P. Quaresma, "Multilingual text classification through combination of monolingual classifiers," in Proceedings of the 4th Workshop on Legal Ontologies and Artificial Intelligence Techniques, vol. 605, pp. 29–38, Citeseer, 2010. 16

[49] J. Novakovic and A. Veljovic, "C-support vector classification: Selection of kernel and parameters in medical diagnosis," in 2011 IEEE 9th international symposium on intelligent systems and informatics, pp. 465–470, IEEE, 2011. 16

[50] F. Colas and P. Brazdil, "Comparison of svm and some older classification algorithms in text classification tasks," in IFIP International Conference on Artificial Intelligence in Theory and Practice, pp. 169–178, Springer, 2006. 16

[51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," The journal of machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014. 17

[52] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998. 18

[53] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science, 1985. 18

[54] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," Neural networks, vol. 18, no. 5-6, pp. 602–610, 2005. 18

[55] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, et al., "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001. 19

[56] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018. 19, 20

[57] T. Pires, E. Schlinger, and D. Garrette, "How multilingual is multilingual bert?," arXiv preprint arXiv:1906.01502, 2019. 20

[58] G. Lample and A. Conneau, "Cross-lingual language model pretraining," arXiv preprint arXiv:1901.07291, 2019. 20

[59] D. M. West, The future of work: Robots, AI, and automation. Brookings Institution Press, 2018. 20

[60] J. Vig, "Bertviz: A tool for visualizing multihead self-attention in the bert model," in ICLR Workshop: Debugging Machine Learning Models, 2019. 23

[61] L. S. Branch, "Consolidated federal laws of canada, criminal code," Sep 2022. 23

[62] H. L. Hammer, M. A. Riegler, L. Øvrelid, and E. Velldal, "Threat: A large annotated corpus for detection of violent threats," in 2019 International Conference on Content-Based Multimedia Indexing (CBMI), pp. 1–5, IEEE, 2019. 24, 32

[63] A. Wester, L. Øvrelid, E. Velldal, and H. L. Hammer, "Threat detection in online discussions," in Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 66–71, 2016. 24

[64] N. Ashraf, R. Mustafa, G. Sidorov, and A. Gelbukh, "Individual vs. group violent threats classification in online discussions," in Companion Proceedings of the Web Conference 2020, pp. 629–633, 2020. 24

[65] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," in Proceedings of the international AAAI conference on web and social media, vol. 11, pp. 512–515, 2017. 32

[66] R. J. Wieringa, Design science methodology for information systems and software engineering. Springer, 2014. 37

[67] D. Stathakis, "How many hidden layers and nodes?," International Journal of Remote Sensing, vol. 30, no. 8, pp. 2133–2147, 2009. 41

[68] S. Karsoliya, "Approximating number of hidden layer neurons in multiple hidden layer bpnn architecture," International Journal of Engineering Trends and Technology, vol. 3, no. 6, pp. 714–717, 2012. 42

[69] M. Madhiarasan and S. Deepa, "Comparative analysis on hidden neurons estimation in multi layer perceptron neural networks for wind speed forecasting," Artificial Intelligence Review, vol. 48, pp. 449–471, 2017. 42

[70] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543, 2014. 43

[71] W. S. Ahmed et al., "The impact of filter size and number of filters on classification accuracy in cnn," in 2020 International conference on computer science and software engineering (CSASE), pp. 88–93, IEEE, 2020. 44

[72] P. Mishra and K. Sarawadekar, "Polynomial learning rate policy with warm restart for deep neural network," in TENCON 2019-2019 IEEE Region 10 Conference (TENCON), pp. 2087–2092, IEEE, 2019. 45, 46

[73] P. Delobelle, T. Winters, and B. Berendt, "Robbert: a dutch roberta-based language model," arXiv preprint arXiv:2001.06286, 2020. 45

[74] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," arXiv preprint arXiv:1907.11692, 2019. 45

[75] H. R. Sofaer, J. A. Hoeting, and C. S. Jarnevich, "The area under the precision-recall curve as a performance metric for rare binary events," Methods in Ecology and Evolution, vol. 10, no. 4, pp. 565–577, 2019. 52

[76] P. A. Lachenbruch, "Mcnemar test," Wiley StatsRef: Statistics Reference Online, 2014. 52

[77] J. Sim and C. C. Wright, "The kappa statistic in reliability studies: use, interpretation, and sample size requirements," Physical therapy, vol. 85, no. 3, pp. 257–268, 2005. 59, 60, 61, 79

[78] U. Reber, "Overcoming language barriers: Assessing the potential of machine translation and topic modeling for the comparative analysis of multilingual text corpora," Communication methods and measures, vol. 13, no. 2, pp. 102–125, 2019.

[79] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in International conference on machine learning, pp. 3145–3153, PMLR, 2017.

[80] J. Risch and R. Krestel, "Toxic comment detection in online discussions," in Deep learning-based approaches for sentiment analysis, pp. 85–109, Springer, 2020.

[81] B. Vidgen, D. Nguyen, H. Margetts, P. Rossini, R. Tromble, K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, et al., "Introducing cad: the contextual abuse dataset," in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 2289–2303, Association for Computational Linguistics, 2021.

[82] A. Herbelot, X. Zhu, A. Palmer, N. Schneider, J. May, and E. Shutova, "Proceedings of the fourteenth workshop on semantic evaluation," in Proceedings of the Fourteenth Workshop on Semantic Evaluation, 2020.

[83] "Germany starts enforcing hate speech law,"

[84] A. L. Wester, "Detecting threats of violence in online discussions," Master's thesis, 2016.

[85] C. Directo Lim, "Detecting legally actionable threats on twitter using natural language processing and machine learning," Master's thesis, 2018.

[86] N. Oostdijk and H. v. Halteren, "N-gram-based recognition of threatening tweets," in International Conference on Intelligent Text Processing and Computational Linguistics, pp. 183–196, Springer, 2013.

[87] N. Oostdijk and H. van Halteren, "Shallow parsing for recognizing threats in dutch tweets," in Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 1034–1041, 2013.

[88] G. Song, D. Huang, and Z. Xiao, "A study of multilingual toxic text detection approaches under imbalanced sample distribution," Information, vol. 12, no. 5, p. 205, 2021.

[89] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," IEEE transactions on Signal Processing, vol. 45, no. 11, pp. 2673–2681, 1997.

[90] E. F. Can, A. Ezen-Can, and F. Can, "Multilingual sentiment analysis: An rnn-based framework for limited data," arXiv preprint arXiv:1806.04511, 2018.

[91] H. Ahn, J. Sun, C. Y. Park, and J. Seo, "Nlpdove at semeval-2020 task 12: Improving offensive language detection with cross-lingual transfer," arXiv preprint arXiv:2008.01354, 2020.

[92] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F. M. R. Pardo, P. Rosso, and M. Sanguinetti, "Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter," in Proceedings of the 13th international workshop on semantic evaluation, pp. 54–63, 2019.

[93] E. Ghadery and M.-F. Moens, "Liir at semeval-2020 task 12: A cross-lingual augmentation approach for multilingual offensive language identification," arXiv preprint arXiv:2005.03695, 2020.

[94] Z. Wang, S. Mayhew, D. Roth, et al., "Cross-lingual ability of multilingual bert: An empirical study," arXiv preprint arXiv:1912.07840, 2019.

[95] S. Wang, J. Liu, X. Ouyang, and Y. Sun, "Galileo at semeval-2020 task 12: Multi-lingual learning for offensive language identification using pre-trained language models," arXiv preprint arXiv:2010.03542, 2020.

[96] M. Pàmies, E. Öhman, K. Kajava, and J. Tiedemann, "Lt@ helsinki at semeval-2020 task 12: Multilingual or language-specific bert?," arXiv preprint arXiv:2008.00805, 2020.

[97] E. W. Pamungkas and V. Patti, "Cross-domain and cross-lingual abusive language detection: A hybrid approach with deep learning and a multilingual lexicon," in Proceedings of the 57th annual meeting of the association for computational linguistics: Student research workshop, pp. 363–370, 2019.

[98] M. Zampieri, P. Nakov, S. Rosenthal, P. Atanasova, G. Karadzhov, H. Mubarak, L. Derczynski, Z. Pitenis, and Ç. Çöltekin, "Semeval-2020 task

12: Multilingual offensive language identification in social media (offenseval 2020)," arXiv preprint arXiv:2006.07235, 2020.

[99] R. Bannink and S. Broeren, "van de looij–jansen pm, de waart fg, raat h. cyber and traditional bullying victimization as a risk factor for mental health problems and suicidal ideation in adolescents," PloS one, vol. 9, no. 4, p. e94026, 2014.

[100] T. K. Ho, "Random decision forests," in Proceedings of 3rd international conference on document analysis and recognition, vol. 1, pp. 278–282, IEEE, 1995.

[101] G. Dunn, Design and analysis of reliability studies: The statistical evaluation of measurement errors. Edward Arnold Publishers, 1989.

[102] P. Brennan and A. Silman, "Statistical methods for assessing observer variability in clinical measures.," BMJ: British Medical Journal, vol. 304, no. 6840, p. 1491, 1992.

[103] I. Tenney, D. Das, and E. Pavlick, "Bert rediscovers the classical nlp pipeline," arXiv preprint arXiv:1905.05950, 2019.

[104] E. Finegan and D. Biber, "Register and social dialect variation: An integrated approach," Sociolinguistic perspectives on register, vol. 315, p. 347, 1994.

[105] T. Gales, "Identifying interpersonal stance in threatening discourse: An appraisal analysis," Discourse Studies, vol. 13, no. 1, pp. 27–46, 2011.

[106] T. Gales, "The stance of stalking: a corpus-based analysis of grammatical markers of stance in threatening communications," Corpora, vol. 10, no. 2, pp. 171–200, 2015.

[107] M. J. Abrams, Uncovering The Genre Of Threatening Texts A multilayered corpus study. Georgetown University, 2019.

[108] A. Beach, "" it's so bomb": Exploring corpus-based threat detection on twitter with discourse analysis," 2019.

[109] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," arXiv preprint arXiv:1601.06733, 2016.

[110] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., "Language models are few-shot learners," Advances in neural information processing systems, vol. 33, pp. 1877–1901, 2020.

[111] E. M. Bender and A. Koller, "Climbing towards nlu: On meaning, form, and understanding in the age of data," in Proceedings of the 58th annual meeting of the association for computational linguistics, pp. 5185–5198, 2020.

[112] N. Kokhlikyan, V. Miglani, M. Martin, E. Wang, B. Alsallakh, J. Reynolds, A. Melnikov, N. Kliushkina, C. Araya, S. Yan, et al., "Captum: A unified and generic model interpretability library for pytorch," arXiv preprint arXiv:2009.07896, 2020.

[113] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," 2017.

[114] J. R. Zilke, E. Loza Mencía, and F. Janssen, "Deepred–rule extraction from deep neural networks," in International conference on discovery science, pp. 457–473, Springer, 2016.

[115] Z. Che, S. Purushotham, R. Khemani, and Y. Liu, "Interpretable deep mod-

els for icu outcome prediction," in AMIA annual symposium proceedings, vol. 2016, p. 371, American Medical Informatics Association, 2016.

[116] I. Arous, L. Dolamic, J. Yang, A. Bhardwaj, G. Cuccu, and P. Cudré-Mauroux, "Marta: Leveraging human rationales for explainable text classification," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 5868–5876, 2021.

Appendices

Appendix A

Systematic Literature Search

Keywords

- Multilingual model
- Cross-learning
- Deep-learning
- Natural Language Processing
- Text classification
- Threat classification
- Threatening text
- Online
- Corpus
- Legal
- Prosecution

Queries & Results

Query	ACM	IEEE	PubMed	arXiv
"Language model" & "Threat classification"	4	0	0	4
"Cross-learning" & "Threat classification"	0	0	0	1
"Deep learning" & "Threat classification"	4	7	1	3
"Language model" & "Threatening text"	1	2	0	0
"Cross-learning" & "Threatening text"	0	1	0	0
"Deep learning" & "Threatening text"	0	0	0	0
"Online" & "Threat classification"	1	7	0	0
"Natural Language Processing" & "Threat classification"	0	0	0	0
"Natural Language Processing" & "Threatening text"	0	0	0	0
"Natural Language Processing" & "Threat classification" & "Online"	0	0	0	0
"Threatening text" & "Corpus"	0	0	0	1
"Threatening text" & "Legal"	1	0	0	0
"Threatening text" & "Prosecution"	0	0	0	0

Table 5: Results of queries per search engine

Inclusion Criteria

- Paper is peer-reviewed (preferably)
- Queried for keyword(s) appear(s) multiple times
- Must help to answer at least one or more research question
- Goal of the research must be clear from abstract
- Date of publication $>$ 2010

Appendix B

Annotation Guidelines

General advice for annotating

- Texts can only fall into two categories: Non-threats and Legally Actionable Threats
- The conceptual basis of 'threatening language' is action. If a verb and/or an expression of intent is used (e.g. 'I will...', 'I want to...', 'I am going to....') or a normative statement that a group, which the speaker is part of, should (e.g. "We should ..." or "Us students should ..."), then it is likely that threatening language is involved.
- Annotations are made on any English and/or Dutch text in the entry, do not translate text which is non-English or non-Dutch. Look out for non-English and non-Dutch terms which are widely used in English or Dutch; such as 'uber' for 'very'/'erg' and 'wunderbar' for 'wonderful'/'fantastisch'.
- Consult sources such as such as Urban Dictionary, Conservapaedia, and Wikipedia. This is particularly important if you encounter terms you are unfamiliar with and you suspect it to be slang.

- Do not click on links in texts.
- Your goal is to apply the coding guidelines presented here. Apply them dispassionately and try to mitigate any personal biases you hold about particular groups.

Codebook

Legally Actionable Threat

The definition of an illegal threat that is used for annotation can be found in Article 285, section 1 of the Dutch Penal Code [29]:

> "Bedreiging met openlijk in vereniging geweld plegen tegen personen of goederen, met geweld tegen een internationaal beschermd persoon of diens beschermde goederen, met enig misdrijf waardoor gevaar voor de algemene veiligheid van personen of goederen of gemeen gevaar voor de verlening van diensten ontstaat, met verkrachting, met feitelijke aanranding van de eerbaarheid, met enig misdrijf tegen het leven gericht, met gijzeling, met zware mishandeling of met brandstichting, wordt gestraft met gevangenisstraf van ten hoogste twee jaren of geldboete van de vierde categorie."

Below a summation is given of all the legally actionable threats presented in Article 285.

- Threat of committing violence with one or more persons in public against one or more persons or their property
- Threat of violence against an internationally protected person [32] or their protected property

- Threat of any crime causing danger to the general security of persons or property or danger to public services (e.g. police, hospitals, etc.)
- Threat of rape or indecent assault [31]
- Threat of any crime against life (i.e. murder)
- Threat of taking hostage
- Threat of aggravated assault
- Threat of arson
- Threat of a terrorist crime

To clarify, indecent assault is defined in the Dutch penal code in Article 246 as follows [31]:

> "Hij die door geweld of een andere feitelijkheid of bedreiging met geweld of een andere feitelijkheid iemand dwingt tot het plegen of dulden van ontuchtige handelingen, wordt, als schuldig aan feitelijke aanranding van de eerbaarheid, gestraft met gevangenisstraf van ten hoogste acht jaren of geldboete van de vijfde categorie."

It is important to note that inelaborate threats of violence or theft are not legally actionable. For example, "I will punch you tomorrow"/"Ik ga je morgen slaan" or "I will steal your bag tomorrow"/"Ik steel je tas morgen" are both not legally actionable.

When the annotator has established a legally actionable threat is present in the text presented to him or her, the annotator should tag the words that signify this. Afterwards, the annotator should annotate the text for containing a legally actionable threat and move on to the next text.

Non-threat

If none of the types of legally actionable threats are found in the text, it is most likely not legally actionable. Below are some examples of threatening languages that are not legally actionable.

- "I will do something to you." The utterer does not threaten with a felony that is listed in Article 285 of the Dutch penal code.
- "I am gonna steal some of your belongings." Threatening with petty theft is not legally actionable.
- "I will punch you tomorrow." Threatening with assault that is not aggravated is not legally actionable.
- "Someone should kill all [group]." This is seen as inciting violence rather than threatening with violence.

As can be seen in the last example, it is not a legally actionable when a speaker incites violence against one or more persons. This is part of a different section of the Dutch penal code. This is not part of this research's scope.

When the annotator has established a text that is not legally actionable for being a threat is present in the text presented to him or her, the annotator should annotate the text for not containing a threat (i.e. non-threat) and move on to the next text.

Appendix C

Classifier architectures

BiLSTM

The hyperparameters of the BiLSTM classifier that were tuned and the final hyperparameters (i.e. text in bold) for the Dutch dataset are:

- Dropout: [0, 0.2, 0.3, 0.4, **0.5**, 0.6, 0.7, 0.8, 0.9, 1]
- Recurrent dropout [0, 0.2, 0.3, 0.4, 0.5, 0.6, **0.7**, 0.8, 0.9, 1]
- Hidden layers [0, **1**, 2, 3]
- Neurons [2, 4, **8**, 16, 32]
- Optimizer [**Adam**, SGD]
- Word embedding dimension: [**50**, 100]
- Sequence length [50, **75**, 100]

The hyperparameters of the BiLSTM classifier that were tuned and the final hyperparameters (i.e. text in bold) for the English dataset are:

- Dropout: [**0**, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
- Recurrent dropout [**0**, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
- Hidden layers [0, **1**, 2, 3]
- Neurons [2, 4, 8, 16, **32**]
- Optimizer [**Adam**, SGD]
- Word embedding dimension: [50, 100]
- Sequence length [50, 75, **100**]

CNN

The hyperparameters of the CNN classifier that were tuned and the final hyperparameters (i.e. text in bold) for the Dutch dataset are:

- Maximum features (i.e. unique tokens): [10000, 20000, 33000]
- Number of filters: [64, **128**, 256]
- Vocabulary size: [**2000**, 4000, 8000, 16000]
- Word embedding dimension: [**50**, 100]
- Sequence length: [50, 75, **100**, 125]
- Learning rate optimizer: [**Adam**, SGD]
- Learning rate scheduler: [**Exponential decay**, Reduce LR on Plateau, Linear Decay]
- Initial learning rate: [1e-2, **1e-3,** 1e-4, 1e-5, 1e-6, 1e-7]

- Learning rate decay rate: [0.5, 0.6, 0.7, 0.8, **0.9**]

The hyperparameters of the CNN classifier that were tuned and the final hyperparameters (i.e. text in bold) for the English dataset are:

- Maximum features (i.e. unique tokens): [10000, 20000, 33000]
- Number of filters: [64, **128**, 256]
- Vocabulary size: [2000, **4000**, 8000, 16000]
- Word embedding dimension: [50, **100**]
- Sequence length: [**50**, 75, 100, 125]
- Learning rate optimizer: [**Adam**, SGD]
- Learning rate scheduler: [**Exponential decay**, Reduce LR on Plateau, Linear Decay]
- Initial learning rate: [1e-2, **1e-3**, 1e-4, 1e-5, 1e-6, 1e-7]
- Learning rate decay rate: [0.5, 0.6, 0.7, 0.8, **0.9**]

BERT (Language Model)

The hyperparameters of the BERT classifier that were tuned and the final hyperparameters (i.e. text in bold) for the Dutch dataset are:

- Initial learning rate: [1e-5, 2e-5, 3e-5, 4e-5, **5e-5**, 5e-6, 5e-7]
-
- Learning rate decay rate: [0.5, 0.6, 0.7, **0.8**, 0.9]

- Warm-up steps: [**250**, 500, 750, 1000]
- Batch size [**4**, 8, 16, 32]
- Gradient accumulation steps [4, **8**, 16]

The hyperparameters of the BERT classifier that were tuned and the final hyperparameters (i.e. text in bold) for the English dataset are:

- Initial learning rate: [**1e-5**, 2e-5, 3e-5, 4e-5, 5e-5, 5e-6, 5e-7]
-
- Learning rate decay rate: [**0.5**, 0.6, 0.7, 0.8, 0.9]
- Warm-up steps: [**250**, 500, 750, 1000]
- Batch size [4, **8**, 16, 32]
- Gradient accumulation steps [4, **8**, 16]

Naive Bayes

The hyperparameters of the NB classifier that were tuned and the final hyperparameters (i.e. text in bold) for the Dutch dataset are:

- Alpha: [0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, **0.2**, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 5, 10]
- Prior fit: [**True**, False]

The hyperparameters of the NB classifier that were tuned and the final hyperparameters (i.e. text in bold) for the English dataset are:

- Alpha: [0, 0.01, **0.02**, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 5, 10]
- Prior fit: [**True**, False]

SVM

The hyperparameters of the SVM classifier that were tuned and the final hyperparameters (i.e. text in bold) for the Dutch dataset are:

- C: [**1**, 5, 10, 25]
- Kernel: [**Linear**, poly, rbf, sigmoid]
- Gamma: [**Auto**, scale]

The hyperparameters of the SVM classifier that were tuned and the final hyperparameters (i.e. text in bold) for the English dataset are:

- C: [1, **5**, 10, 25]
- Kernel: [**Linear**, poly, rbf, sigmoid]
- Gamma: [**Auto**, scale]

Appendix D

Performance metrics on Dutch threat corpus

F1-scores

Fold	BiLSTM	CNN	LM	NB	SVM
1	0.774	0.967	0.988	0.94	0.966
2	0.706	0.976	0.987	0.957	0.974
3	0.763	0.98	0.985	0.945	0.968
4	0.771	0.972	0.987	0.938	0.956
5	0.73	0.978	0.988	0.947	0.972
6	0.739	0.979	0.99	0.948	0.969
7	0.758	0.972	0.988	0.95	0.964
8	0.772	0.972	0.99	0.946	0.968
9	0.741	0.973	0.991	0.94	0.968
10	0.782	0.982	0.985	0.943	0.972
Mean	0.736	0.975	**0.988**	0.945	0.967
Std	0.023	0.001	**0.000**	0.001	0.004

Table 6: F1 Macro scores per classifier per fold

Fold	BiLSTM	CNN	LM	NB	SVM
1	0.948	0.993	0.998	0.989	0.993
2	0.939	0.995	0.997	0.992	0.995
3	0.951	0.996	0.997	0.990	0.994
4	0.951	0.994	0.997	0.989	0.991
5	0.944	0.996	0.998	0.990	0.994
6	0.94	0.996	**0.999**	0.990	0.994
7	0.948	0.994	0.998	0.990	0.993
8	0.950	0.994	0.998	0.990	0.994
9	0.947	0.994	0.998	0.989	0.997
10	0.953	0.996	0.997	0.990	0.997
Mean	0.947	0.995	**0.998**	0.990	0.994
Std	0.004	0.001	**0.000**	0.001	0.002

Table 7: F1 Micro scores per classifier per fold

Fold	BiLSTM		CNN		LM		NB		SVM	
	0	1	0	1	0	1	0	1	0	1
1	0.972	0.577	0.996	0.937	0.999	0.977	0.994	0.886	0.996	0.935
2	0.967	0.444	0.997	0.954	0.999	0.975	0.996	0.918	0.997	0.95
3	0.974	0.551	0.998	0.962	0.998	0.971	0.995	0.895	0.997	0.94
4	0.974	0.567	0.997	0.947	0.999	0.976	0.994	0.881	0.995	0.916
5	0.971	0.500	0.998	0.957	0.999	0.976	0.995	0.900	0.998	0.946
6	0.968	0.509	0.998	0.960	0.999	0.981	0.995	0.902	0.997	0.941
7	0.972	0.544	0.998	0.947	0.999	0.976	0.995	0.903	0.997	0.931
8	0.973	0.571	0.997	0.946	0.999	0.982	0.995	0.898	0.996	0.94
9	0.972	0.51	0.997	0.949	0.999	0.984	0.994	0.884	0.997	0.939
10	0.975	0.589	0.998	0.966	0.999	0.972	0.994	0.892	0.997	0.947
Mean	0.972	0.536	0.998	0.953	**0.999**	**0.977**	0.900	0.914	0.997	0.939
Std	0.002	0.043	0.001	0.008	**0.000**	**0.004**	0.001	0.010	0.001	0.009

Table 8: F1 Minority and Majority scores per classifier per fold

PR-AUC scores & graphs per classifier

Fold	BiLSTM	CNN	LM	NB	SVM
1	0.596	0.97	0.978	0.895	0.969
2	0.459	0.979	0.976	0.928	0.969
3	0.563	0.941	0.972	0.917	0.959
4	0.581	0.961	0.977	0.928	0.969
5	0.503	0.948	0.977	0.927	0.927
6	0.531	0.947	0.982	0.914	0.915
7	0.558	0.947	0.977	0.947	0.968
8	0.587	0.979	0.983	0.896	0.958
9	0.523	0.958	0.984	0.908	0.938
10	0.6	0.959	0.972	0.915	0.959
Mean	0.550	0.954	**0.978**	0.903	0.940
Std	0.043	0.008	**0.004**	0.009	0.009

Table 9: PR-AUC scores per classifier per fold

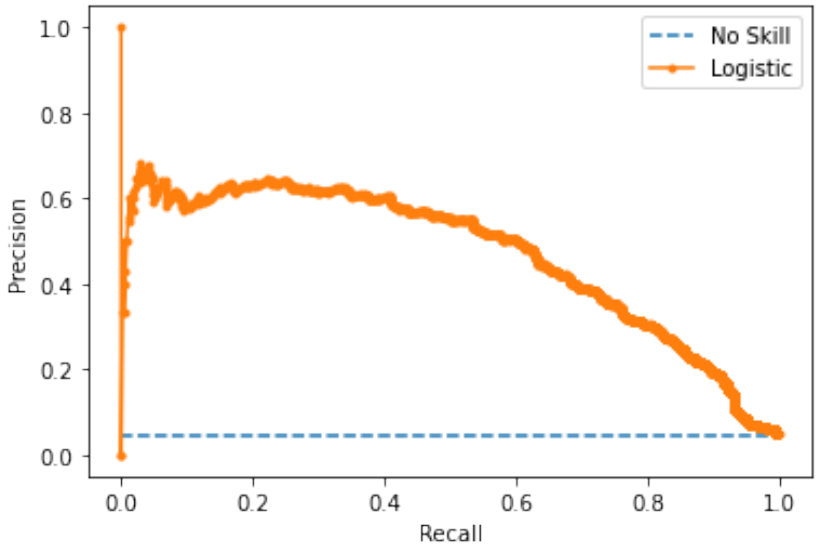

Figure 14: PR-AUC score of BiLSTM classifier

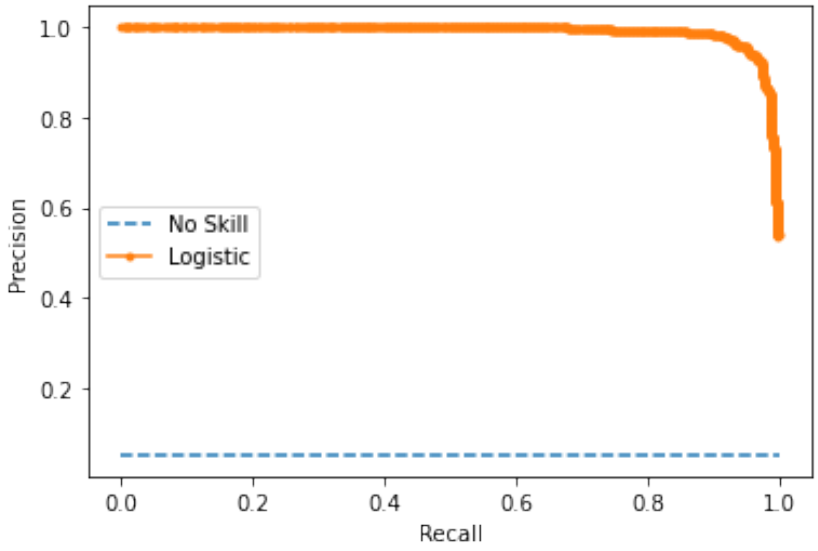

Figure 15: Most representative PR-AUC graph of CNN classifier

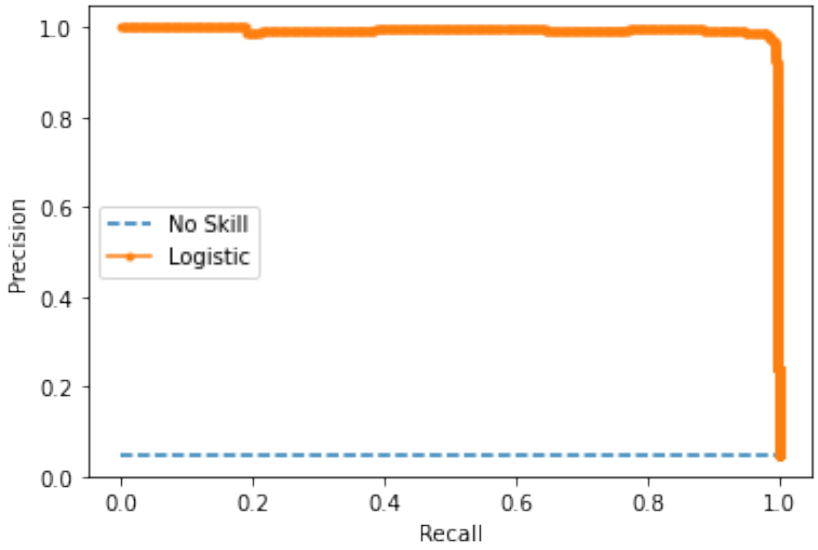

Figure 16: Most representative PR-AUC graph of LM classifier

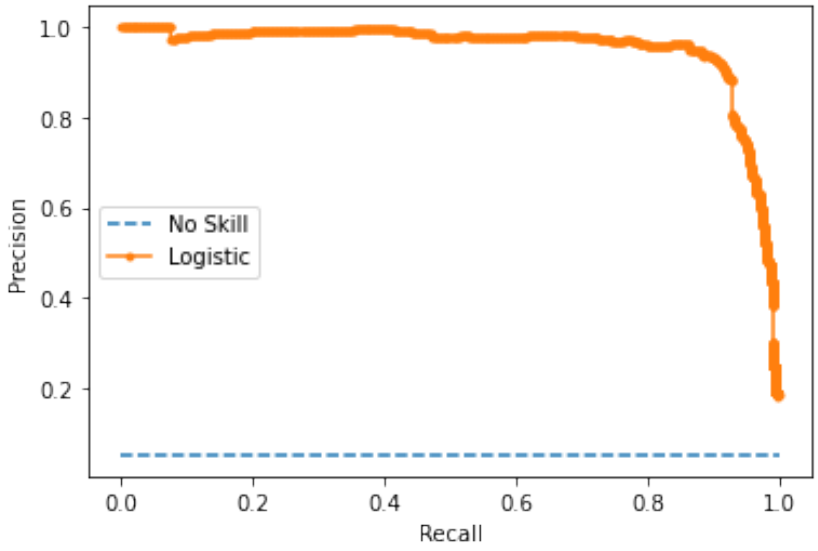

Figure 17: Most representative PR-AUC graph of Naive Bayes classifier

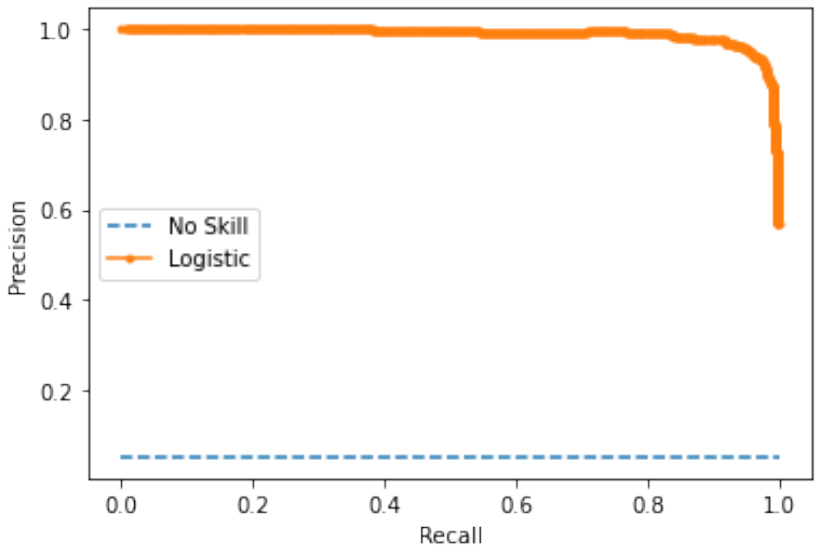

Figure 18: Most representative PR-AUC graph of SVM classifier

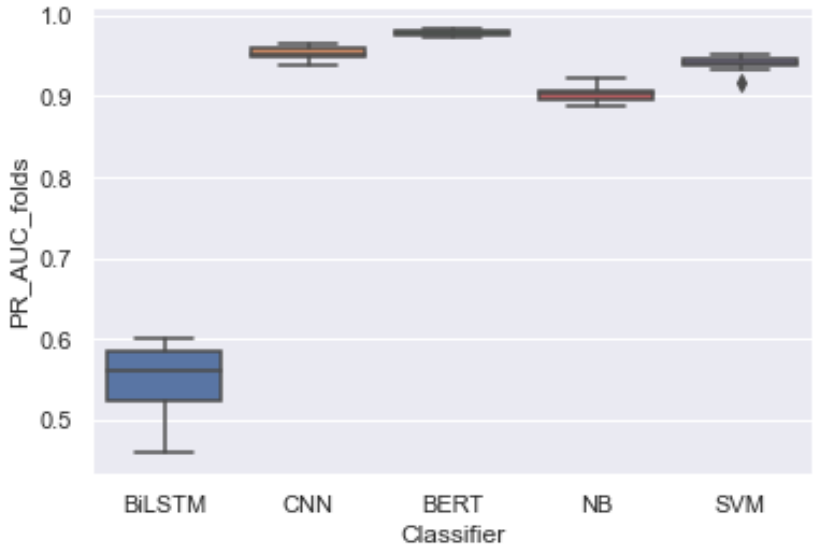

Figure 19: Boxplot of PR-AUC scores per classifier

Appendix E

Performance metrics on English threat corpus

F1-scores

Fold	BiLSTM	CNN	LM	NB	SVM
1	0.717	0.661	0.893	0.551	0.640
2	0.689	0.652	0.905	0.584	0.606
3	0.761	0.769	0.919	0.628	0.776
4	0.696	0.742	0.906	0.662	0.742
5	0.822	0.828	0.891	0.604	0.747
6	0.840	0.893	0.914	0.799	0.877
7	0.871	0.911	0.981	0.743	0.874
8	0.850	0.907	0.982	0.780	0.878
9	0.874	0.859	0.984	0.774	0.877
10	0.880	0.902	0.980	0.738	0.832
Mean	0.800	0.812	**0.936**	0.686	0.785
Std	0.073	0.095	**0.039**	0.086	0.096

Table 10: F1 Macro scores per classifier per fold

Fold	BiLSTM	CNN	LM	NB	SVM
1	0.925	0.893	0.982	0.923	0.896
2	0.928	0.923	0.980	0.933	0.900
3	0.972	0.972	0.985	0.966	0.972
4	0.970	0.970	0.983	0.967	0.97
5	0.989	0.979	0.980	0.964	0.971
6	0.980	0.985	0.981	0.976	0.981
7	0.983	0.987	0.981	0.972	0.981
8	0.981	0.986	0.982	0.974	0.983
9	0.983	0.981	0.984	0.973	0.984
10	0.981	0.986	0.981	0.971	0.978
Mean	0.969	0.966	**0.982**	0.962	0.962
Std	0.022	0.030	**0.002**	0.017	0.032

Table 11: F1 Micro scores per classifier per fold

Fold	BiLSTM		CNN		LM		NB		SVM	
	0	1	0	1	0	1	0	1	0	1
1	0.959	0.474	0.941	0.380	0.991	0.795	0.96	0.143	0.943	0.338
2	0.961	0.414	0.960	0.345	0.990	0.820	0.965	0.203	0.946	0.266
3	0.986	0.536	0.986	0.551	0.992	0.846	0.983	0.272	0.986	0.567
4	0.984	0.410	0.985	0.500	0.991	0.820	0.983	0.340	0.985	0.5
5	0.990	0.656	0.990	0.667	0.990	0.793	0.981	0.227	0.985	0.51
6	0.990	0.689	0.992	0.794	0.989	0.838	0.987	0.610	0.990	0.873
7	0.991	0.750	0.993	0.829	0.990	0.809	0.986	0.500	0.990	0.719
8	0.990	0.710	0.993	0.822	0.990	0.815	0.988	0.571	0.991	0.765
9	0.991	0.758	0.990	0.727	0.990	0.836	0.986	0.561	0.992	0.762
10	0.990	0.769	0.993	0.811	0.991	0.824	0.985	0.490	0.988	0.677
Mean	0.983	0.617	0.982	0.643	**0.990**	**0.820**	0.980	0.392	0.980	0.598
Std	0.012	0.137	0.017	0.177	**0.001**	**0.017**	0.009	0.165	0.018	0.186

Table 12: F1 Minority and Majority scores per classifier per fold

PR-AUC scores & graphs per classifier

Fold	**BiLSTM**	**CNN**	**LM**	**NB**	**SVM**
1	0.581	0.527	0.830	0.161	0.443
2	0.477	0.392	0.848	0.219	0.329
3	0.626	0.622	0.865	0.600	0.621
4	0.644	0.605	0.849	0.619	0.605
5	0.754	0.743	0.824	0.582	0.630
6	0.756	0.818	0.858	0.692	0.769
7	0.796	0.845	0.832	0.680	0.763
8	0.769	0.831	0.849	0.688	0.789
9	0.791	0.760	0.856	0.662	0.816
10	0.774	0.831	0.856	0.644	0.714
Mean	0.697	0.697	**0.847**	0.555	0.648
Std	0.103	0.146	**0.013**	0.186	0.151

Table 13: PR-AUC scores per classifier per fold

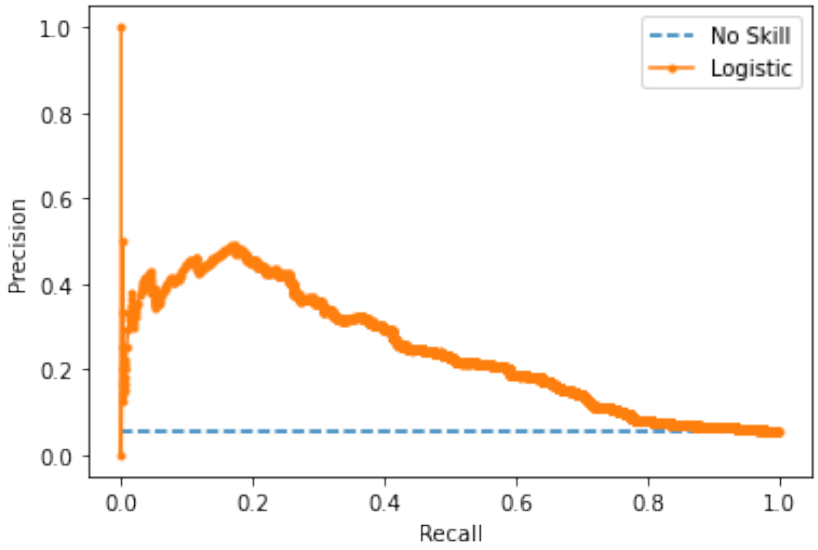

Figure 20: PR-AUC score of BiLSTM classifier

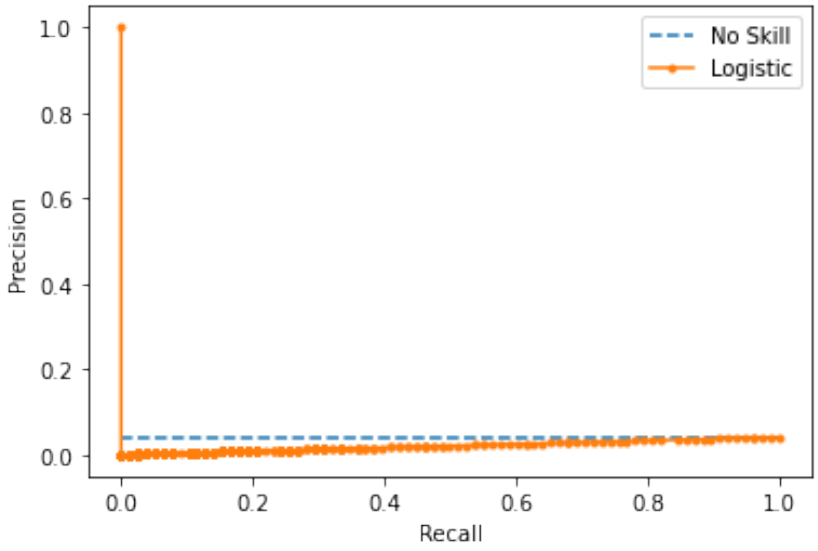

Figure 21: Most representative PR-AUC graph of CNN classifier

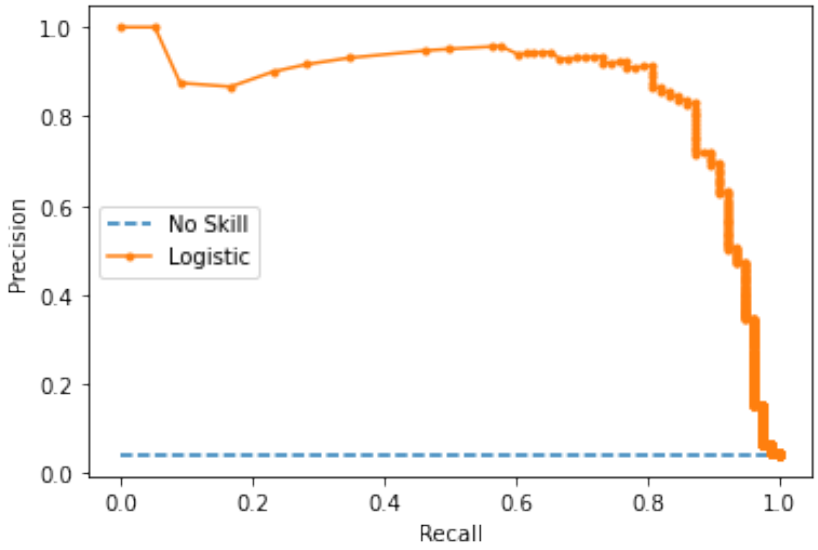

Figure 22: Most representative PR-AUC graph of LM classifier

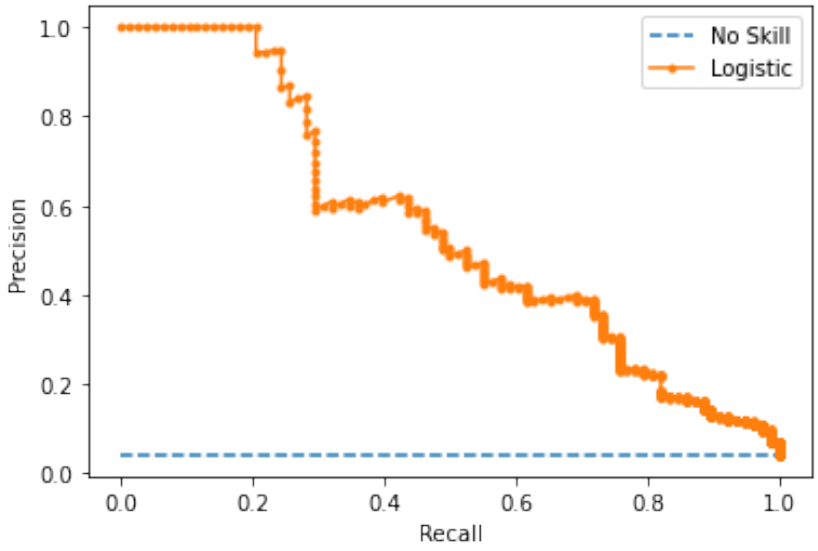

Figure 23: Most representative PR-AUC graph of Naive Bayes classifier

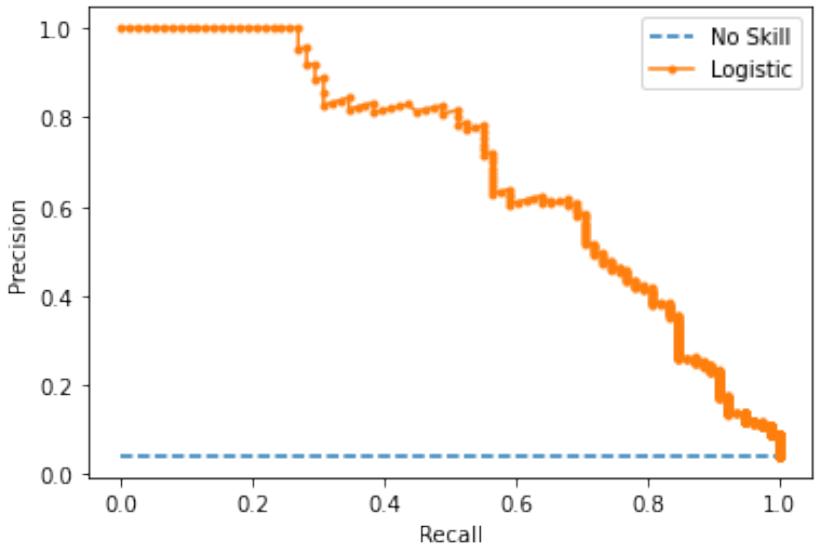

Figure 24: Most representative PR-AUC graph of SVM classifier

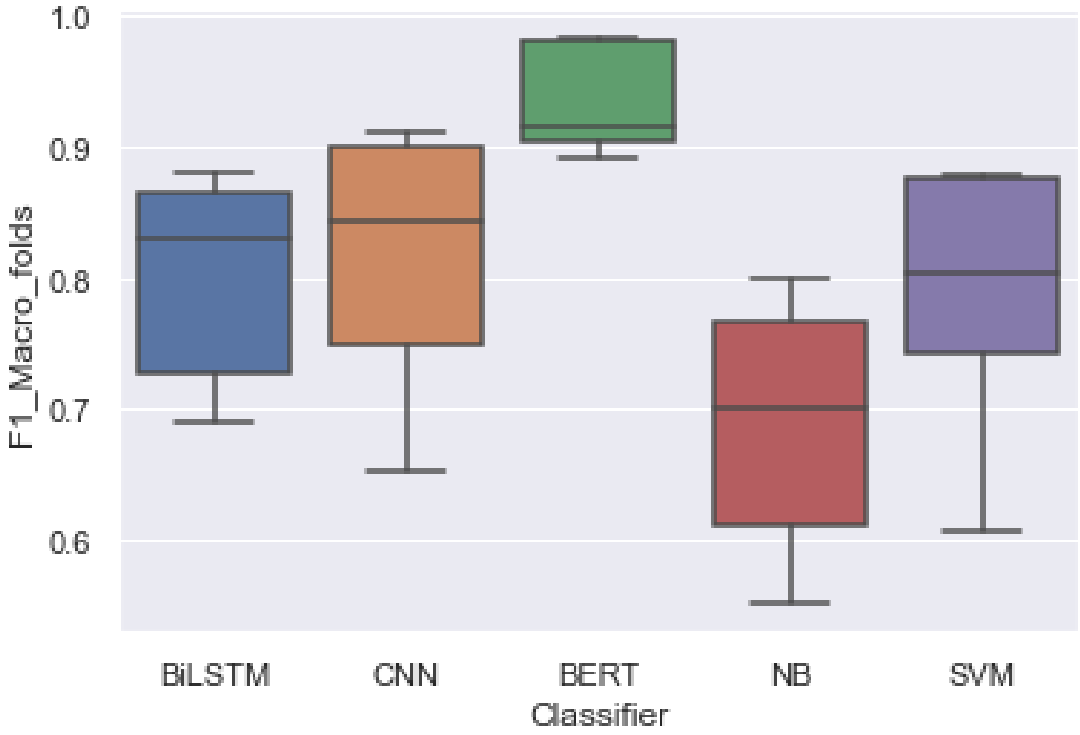

Figure 25: Boxplot of PR-AUC scores per classifier

Appendix F

Performance metrics on internal police data

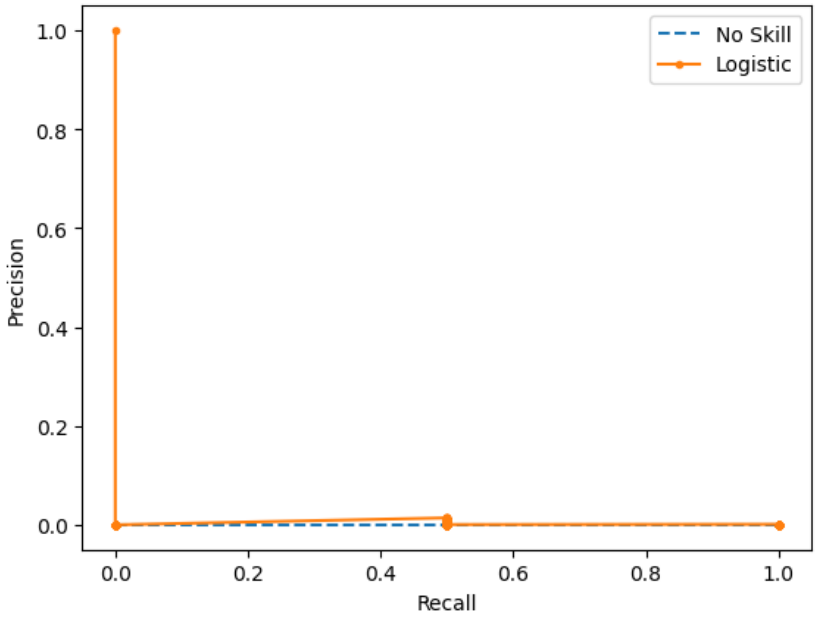

Figure 26: PR-AUC score of BiLSTM classifier

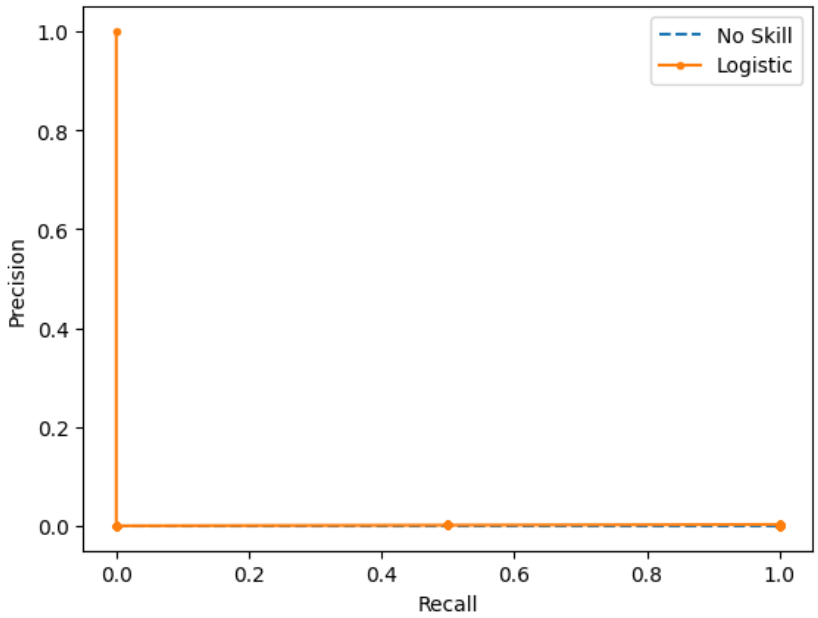

Figure 27: PR-AUC score of CNN classifier

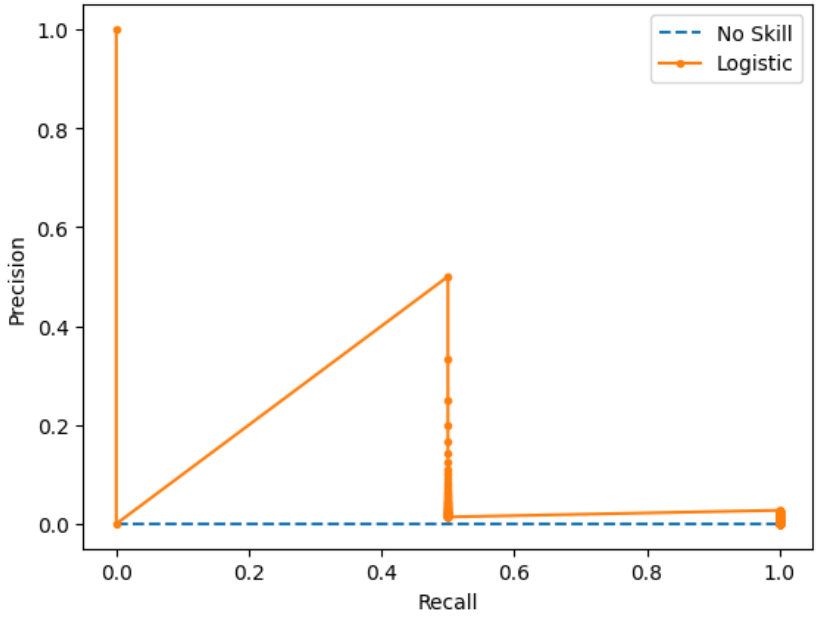

Figure 28: PR-AUC score of LM classifier

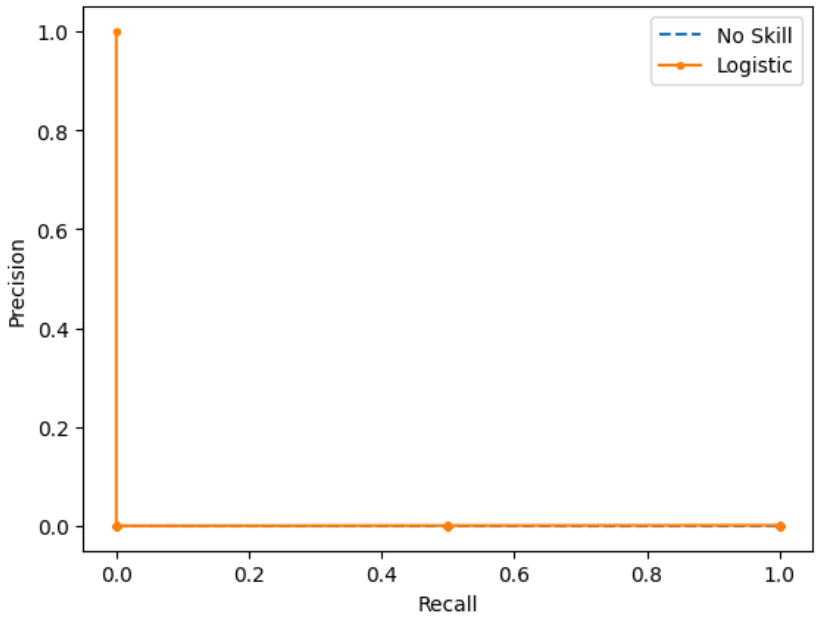

Figure 29: PR-AUC score of Naive Bayes classifier

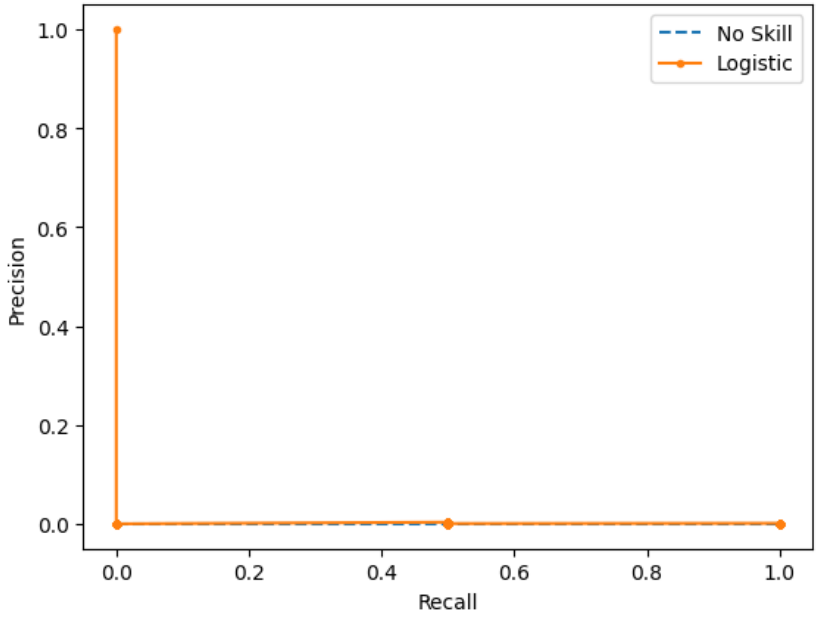

Figure 30: PR-AUC score of SVM classifier

Appendix G

XAI examples of classifications on the Dutch dataset

Correct classifications

Figure 31: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

Figure 32: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

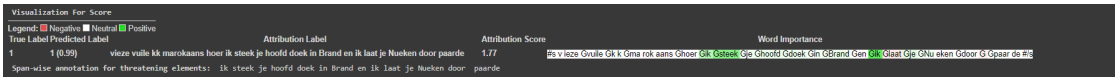

Figure 33: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

Figure 34: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

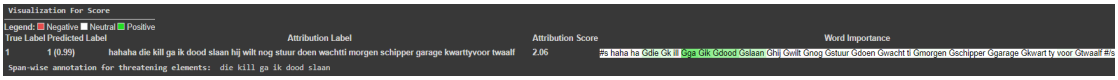

Figure 35: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

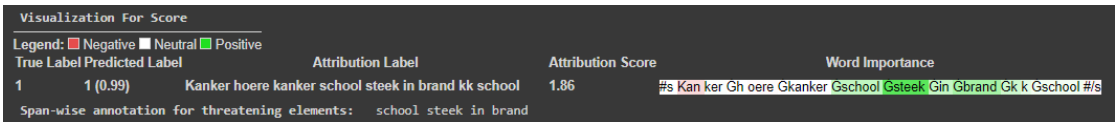

Figure 36: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

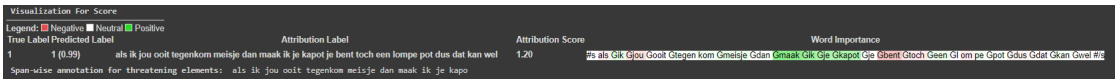

Figure 37: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

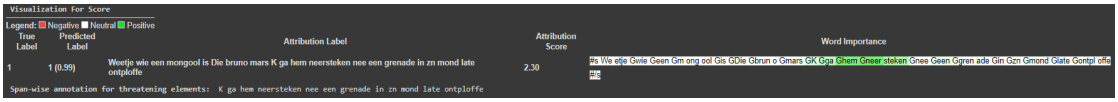

Figure 38: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

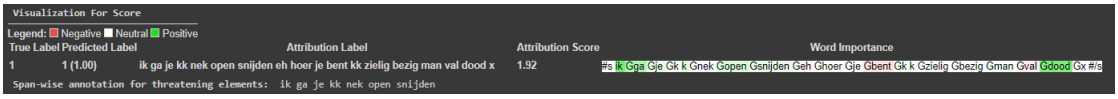

Figure 39: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

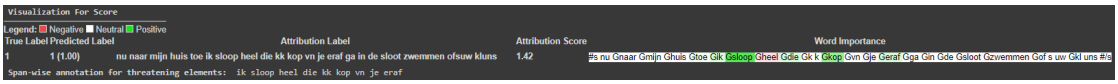

Figure 40: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

Incorrect classifications

Figure 41: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

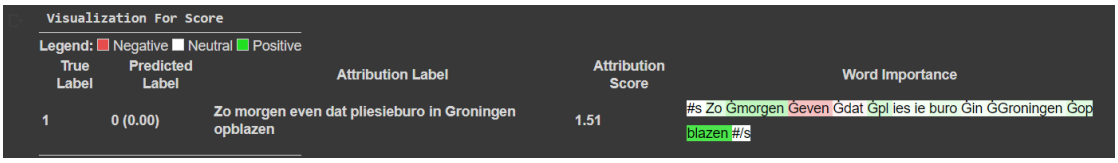

Figure 42: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

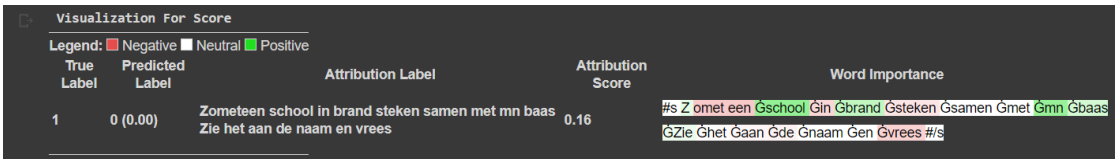

Figure 43: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

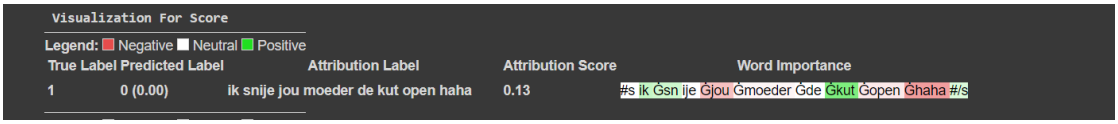

Figure 44: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

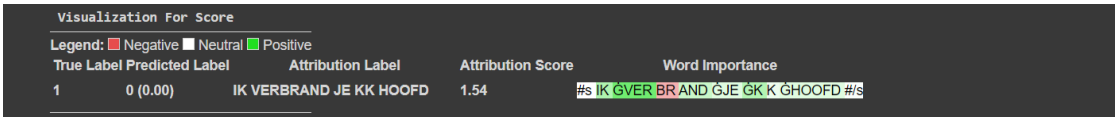

Figure 45: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

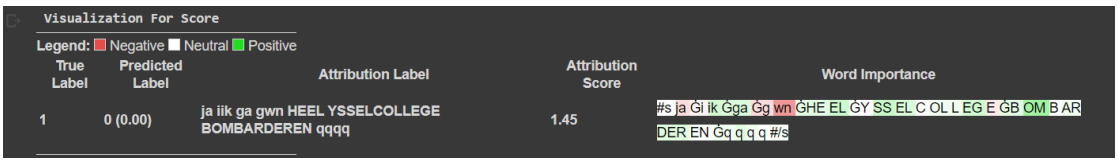

Figure 46: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

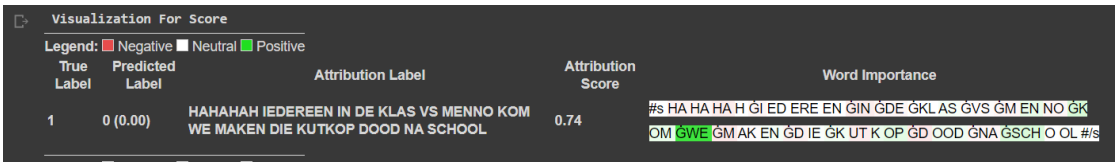

Figure 47: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

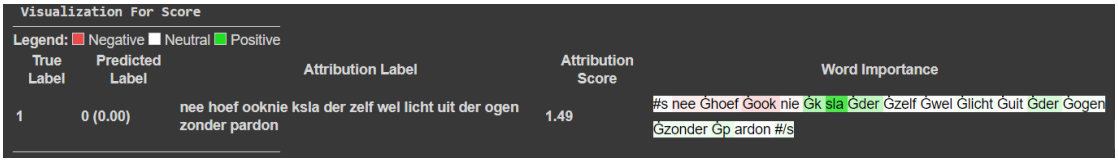

Figure 48: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

Figure 49: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher

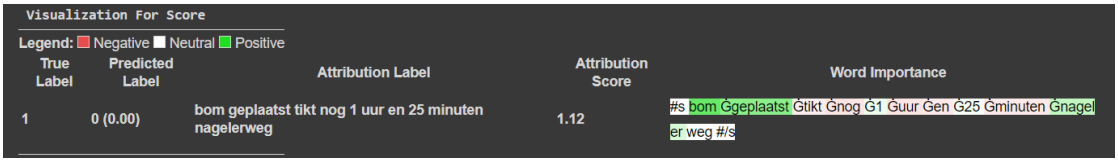

Figure 50: Visualization of estimated feature relevance alongside the span-wise annotation of the researcher