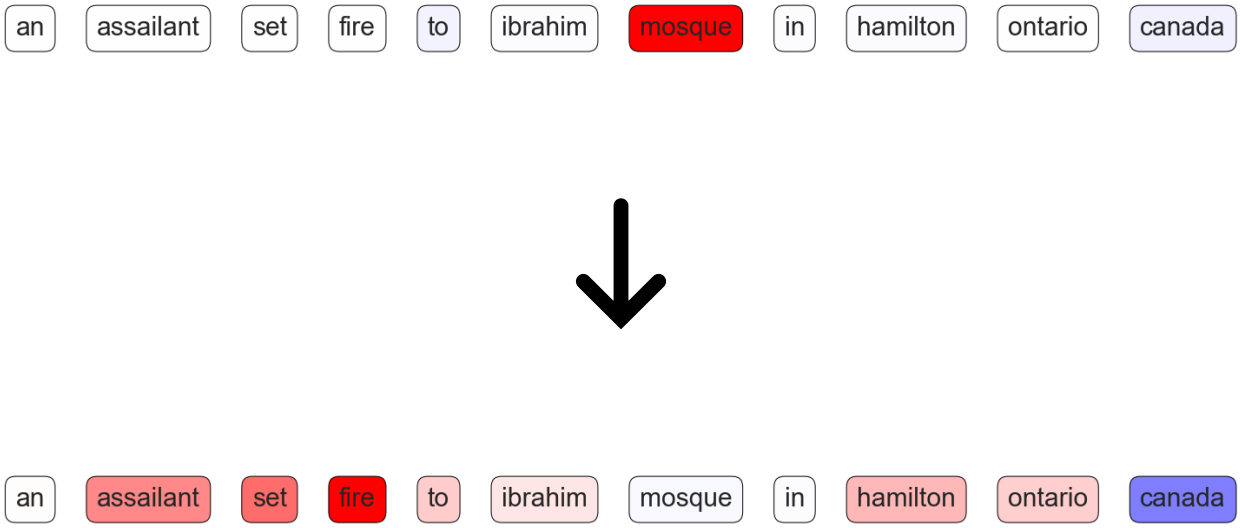
CHALMERS
UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG



Informed regularization

Aiding the identification of spurious correlations

Master's thesis in Computer science and engineering

Fredrik Möller

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2020

Master's thesis 2020

Informed regularization

Aiding the identification of spurious correlations

Fredrik Möller

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2020

Informed regularization
Aiding the identification of spurious correlations
Fredrik Möller

Advisor: Richard Johansson, Department of Computer Science and Engineering
Supervisor: Johan Larsson Hörkén, Recorded Future
Examiner: Richard Johansson, Department of Computer Science and Engineering

Master's Thesis 2020
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: LRP result for a sentence before and after word filter application

Typeset in LaTeX
Gothenburg, Sweden 2020

Informed regularization
Aiding the identification of spurious correlations
Fredrik Möller
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Today, end-to-end neural networks that feature deep and complex architectures, are common tools to use in natural language processing. By using these methods it has become harder to identify which inputs have contributed the most to a model's classification. This issue leads to the problem of models overfitting on features that cannot directly be identified by a developer.

To open up the black box of complex deep learning natural language processing systems, this study aims to investigate what information can be extracted from the data used to train a model and how the model's inputs are weighted during prediction. This thesis aims to present methods that can aid in the identification of differences between the population a developer intends to model with a data set and what correlations a model makes from the true content of the data.

By presenting three novel methods that can aid a developer with the task of identifying spurious correlations, it was possible to present information regarding a spurious correlation between two pre-selected keywords and a model's classification. It was also shown that the identification and reduction of spurious correlations is a tricky subject. Results showed that, from the reduction of the spurious correlation associated with the selected keyword, the model made another correlation which could be considered as spurious.

Keywords: NLP, Explainability, Regularization, Layer-wise relevance propagation, TF-IDF, NCOF.

Acknowledgements

I would like to express my gratitude for the time and support I have received from my examiner Richard Johansson, at Chalmers University of Technology, and my supervisor Johan Larsson Hörkén from Recorded future. I would also like to extend a thank you to Danila Petrelli, linguistics manager at Recorded Future, for the linguistic expertise you have provided, and Staffan Truvé, CTO and co-founder of Recorded Future, for the feedback you have given me during the project.

Fredrik Möller, Gothenburg, December 2020

Contents

List of Figures

List of Tables

Nomenclature

CNN Convolutional neural network
EDA Easy data augmentation
IDF Inverse document frequency
LRP Layer-wise relevance propagation
NLP Natural language processing
NN Neural network
OOV Out-of-vocabulary
$TF\text{-}IDF$ Term frequency-inverse document frequency

1
Introduction

Natural language processing (NLP) has advanced tremendously in the past years and so have the commonly used methods [2]. Today end-to-end neural networks (NN), featuring deep and complex architectures, are common. The shift from traditional feature-rich and statistical methods to neural networks has moved the linguistic knowledge from the algorithms into the data used to train the networks. This shift have resulted in that the decision process of the NLP models have become harder to understand [3]. The shift to neural networks has also made the identification and separation of spurious and true correlations in a model's training data more complex, given the vast amount of data required to train NN.

The issue of separating spurious and true correlations form data and its labels is not a trivial problem when the underlying model is unknown. This problem is not unique to the NLP field and has been investigated in control theory by Liu, Shah, and Jiang among others [4]. In their paper "On-line outlier detection and data cleaning" Liu, Shah, and Jiang mention that it is generally hard to filter outliers while simultaneity keeping track of a changing process model [4]. In other words, tracking what the data models while removing samples is difficult since what is considered as a true correlation in the data also changes when samples are removed. Since this tracking is practically impossible, it must be accepted that when removing data with the intent of reducing spurious correlation there is a risk that true correlations are removed leading to an alteration of what the data models. Since an alteration cannot be measured directly from the data the change needs to be estimated from other sources such as measuring how the data influences an NLP model's predictions.

Neither of the methods that will be investigated and used in this thesis are able to separate a spurious from a true correlation. As the issue of correlation separation remains unsolved, this thesis will focus will focus on presenting information regarding the data and how it has influenced a model such that a human can make an informed decision regarding what is to be considered as a true or spurious correlations.

An example of a method that can be used to investigate the importance of the inputs of an NLP model is Layer-wise relevance propagation (LRP) [5]. The LRP method acts by propagating the prediction of a model back to its inputs, further theory regarding the method is presented in Chapter 2, Section 2.4.1. In this study three novel ways of identifying possible spurious correlations in a model and the data used to train it will be presented. The LRP method will be used both as a

method for aiding bias identification but also as a way to show and evaluate if it is possible to reduce the importance of identified words for a model's prediction.

1.1 Relation between bias and spurious correlations

Bias will in this study refer to the phenomena of a model producing results that are systematically incorrect due to spurious correlations. These correlations would entail that a model heavily bases its prediction on entities that do not hold a true correlation between the input, its label, and the subject being modelled. What is considered to be a spurious correlation can be considered a subjective matter and depends on both one's own interpretation of what information is present in the data and the classification task.

It can also be noted that due to an ever-changing social landscape building a predictive model from historical data will always carry the risk of inheriting a spurious correlation [6].

In Section 2.1 further discussion regarding bias and the implications of bias in a system is held.

1.2 Purpose and aim

The task of this project is to investigate the possibility of producing informed support regarding possible spurious correlation in an NLP model by presenting information regarding the models prediction and the data it has been trained on.

The hypothesis that will be tested is that if a spurious correlation is identifiable from the predictions of a model, then the correlation should also be identifiable in the data or the weighting of the model's inputs. In conjunction to this hypothesis it will be investigated if the spurious correlation can be reduced or removed by applying a focused regularization method to the data.

This thesis aims to present methods that can aid a developer to identify possible spurious correlations. The aid shall be in the form of concise information regarding content in the data that is either skewed in its class distribution or have a high weighting in in a model for incorrect classifications

The end goal of this study is to develop methods that can support a developer with the process of detecting content in a data set which affect a model's classification disproportionately to its frequency in the data to train a model. This study intends to not focus on a specific type of content but instead present objective re-

sults regarding how the distribution of information between the classes in a data set has affected its weighting in the model's prediction.

1.3 Objectives

To reach the aim and goals presented in Section 1.2 this thesis will investigate the following questions:

- What information should be presented to a developer such that an informed decision can be made regarding if a spurious correlation have been identified?
- What methods can be used to produce this information?
- How is the presented information weighted in a model during classification?
- Are the methods sufficient to replace current processes for the identification of spurious correlations?

This thesis will focus to reach the proposed objectives through filtering identified keywords that could be sources of spurious correlations. This approach will be a form of regularization of the data and will act as a method to reduce the models weighting of the identified keywords.

1.4 Disclaimers and project-wide limitations

This study focuses first and foremost on the explainability aspect of identification and reducing biases in an NLP model. This is due to the project being conducted as a proof of concept of the presented techniques and methods to show their potential for reducing the impact of specific words in a model's classification. For this reason, the idea of only calling the project a success if and only if the accuracy of a model is increased when the data is regularized using the proposed methods is deemed as wrong.

This project does not aim to conduct an extensive study regarding the effectiveness of different convolutional neural network(CNN) model structures. This limitation also includes extensive hyperparameter sweeps. The reason for this limitation is the increase in complexity and computational cost associated with training and evaluation of CNN models with more parameters than the one used in this study. Another cause for this limitation is the cost of performing LRP for larger models and the challenge of implementing the method on multiple models.

1.5 Related work

The importance of being aware of what a model bases its predictions on was shown by Kiritchenko and Mohammad in their study from 2018 [7]. In their study of 219 sentiment analysis systems over 75% of the evaluated systems scored sentences differently depending on which gender or race the sentence involved. They also found

that a bias was more common for race than for gender, with score differences up to 34% for sentences containing African American versus European American names. Thus indicating the prevalence and effect spurious correlations can have in sentiment analysis systems.

Another example of the importance of understanding the predictions of an NLP model is Amazon's attempt to create a classifier to score job applications sent to them [8]. It was found that the classifier scored male applicants consistently higher than females. The reason for this was that the data Amazon used to train the model consisted mainly of applications sent by males [8]. The data they used was not faulty per se, but the data they used was unrepresentative of the population they were trying to model.

The approach of removing words to reduce overfitting is not new. Søgaard and Johannsen presented in their paper "Robust learning in random subspaces: equipping NLP for OOV effects" that by dropping features from the data it was possible to reduce the classification error rate in a model for data sampled from a previously unseen domain [9]. The difference between the method presented by Søgaard and Johannsen and the approaches this thesis will present is that this project aims to focus on overrepresentations between the classes in the data, while Søgaard and Johannsen focused on reducing the effect of out-of-vocabulary (OOV) effects.

NLP models are very prone to overfitting and generally lose performance quite quickly when the training and test data are pooled from two separate domains and are not representative of each other [9]. One cause for this quick degradation of the model's performance is called OOV effects [9]. A high-level explanation of OOV effects is that the model has made correlations between the labels of the training data and the specific language used in the subset. These correlations cause the model to be unable to make good generalisations such that the model misclassify parts of the test data. What Søgaard and Johannsen presented in their paper, [9], was that by reducing the effect of OOV words the accuracy of a model could be increased, thus reducing the impact of spurious correlations created by the OOV words. Where Søgaard and Johannsen investigated the impact of OOV words this thesis aims to investigate the influence of overrepresented words in subsets of the training data in the context of explainability and identification of spurious correlations.

Research that presents methods for identifying and correcting a spurious correlation between a model's training data and its labels have been conducted by Jiang and Nachum [10]. Jiang and Nachum showed that by re-weighting the class distribution in the training data the impact of a manually induced bias in the data could be reduced. Their presented method for re-weighting the data distribution in the training data does however not take into account the models weighting of specific content during classification for the re-weighting process.

LRP has been applied to NLP classification tasks with good results regarding explaining model predictions [11, 12], and previous work has identified the issue with

bias in NLP models and proposed methods on how to identify them [13, 14], no past papers have been found to use bias identification methods as the base for data regularization.

As the LRP method is possible to use as the base for a heat map of a model's input it is important to note that previous efforts have been made to extract information and interpret heat maps. Samek *et al* [15] mention that solution of how to effectively quantify and measure the quality of a heatmap was at the time, 2017, an open question.

Augmentation of the training data to reduce overfitting and skewed weighting when working with neural networks is very common, and research is performed in this field. An example of this is Jason Wei and Kai Zou´s [16] method easy data augmentation (EDA). They showed that their proposed techniques for data augmentation had a positive effect on model performance on both large and small data sets. However, their proposed techniques are stochastic operators on the data and do not present a developer the opportunity to interact with the methods other than tuning hyperparameters. This study, in contrast to the work presented by Wei and Zou, will work towards augmenting the data in a more deterministic fashion, a move away from stochastic operators deciding what data is to be augmented. Jason Wei and Kai Zou also mentions that there is a lack of standardised data augmentation techniques in NLP, in contrast to the field of computer vision.

An idea of how to regularize a neural network as a measure to counteract the issue of overfitting was presented by Srivastava *et al* in their paper "Dropout: A Simple Way to Prevent Neural Networks from overfitting" [17]. Their presented method of regularization was to deactivate random hidden neurons in the network during training, this method was proven to significantly reduce overfitting by preventing the hidden units from co-adapting too much. Although Srivastava *et al* focus on the regularizing a neural network through its parameters and this study aims to regularize a model by altering its training data. The idea of reducing the impact of specific neurons or features in the data to increase the a model's ability to generalise is common to Srivastava´s *et al* paper and this thesis.

2
Theory

This chapter will present the theory and related work that is relevant to the scope of the thesis. The chapter will begin by explaining the notion of biases in data sets and the implication of their presence. It is then followed by theory related to the type of model used for the NLP task, a CNN set in a supervised learning environment, together with theory related to how text is represented for solving the classification task. The two final sections present the theory regarding how the task of identifying possible spurious correlations in the chosen model and the training data can be solved.

2.1 Overfitting and bias relationships

A spurious correlation generally occurs when two variables are statistically related but not directly causally related. Meaning that they falsely appear to share a relation to each other. The underlying data rather than the chosen algorithm are often the root cause of a spurious correlation. This type of correlations and biases can be introduced to the data from the sample selection process or from oversampling from specific populations, this can cause a data set to become misrepresentative of its intended population [18]. Spurious correlations can also be introduced into data through user generated feedback loops, where users may engage with content in an unintended way which can cause biases in a system to strengthen [19].

Because spurious correlations have a detrimental effect on a models accuracy and classification robustness they are often first noted when a model have been trained. Spurious correlations can be avoided to an extent by identifying some of the possible sources of bias in a models training data. One type is called sample bias, it appears when data used to train a model is unrepresentative of the domain the model will operate in. This type of bias is reduced by expanding the domain of the training data such that it represents the models operational domain [20]. Another type of data bias that is not unusual is called prejudice bias. This bias is the result of including and training a model on data influenced by cultural, ethnic, or other stereotypes [20].

It is possible to conceptualise that if there is a discrepancy between a model's accuracy on some training and test data there must exist a difference between the populations that the sets tries to model. If we assume that the sets are supposed to be representative of each other then the difference in modelled populations can be attributed to spurious correlations in the data sets. An example such correlation

can be the difference in the distribution of information present in a data set between what the creator of the data set intended and the real distribution the data contains. Thus it is possible for the spurious correlations in the training data to affect a model during its training and cause it to lose accuracy when classifying the test data set due to the two sets being unrepresentative of each other.

Another general limitation that should be kept in mind is that the data available to train a model is always finite, thus a perfect representation of reality can never be created. Due to the issue of finite data, there will always be a discrepancy between the choice of data used to train a model and its representation of the population that is being modelled.

2.1.1 Creating bias in a data set

By recalling the notion from the previous section, 2.1, that data is always finite and that a perfect representation of a population can never be created. It can be noted that this discrepancy can thus be used as a source of biases by using the limited data to intentionally represent a population it does not fairly model. Another method to create a bias in a data set is to skew a subset of the data towards one class, such that identifying features of the subset are overrepresented in that class.

By consciously creating a bias towards a few select keywords in a data set, and through training inducing this bias in a model, it is possible to use the knowledge of the bias to evaluate the ability of different methods to detect it and investigate how it can be reduced. This manually created bias can be considered as a spurious correlation due to not being naturally present in the data. As natural spurious correlations can be difficult to detect due to the subjective matter of identifying the difference between what is being modelled by the data and what is intended to be modelled, using a data set with a manually created bias as a test environment for the identification of said bias creates the possibility of working with a data set with a known issue.

2.2 Supervised learning and neural network classifiers

Supervised learning tasks are a subset of problems in machine learning. A supervised learning task can be generalised as an optimisation problem with the goal of optimising the parameters of a NN such that a loss function, $L(x, y)$, is minimised, a general example of a loss function is presented in Equation 2.1. Gradient descent is an algorithm that is often used to update the weights of a NN during its training to minimise the networks loss function, such that the optimal prediction f^* can be reached [21]. The algorithm finds the minimum of the loss function by taking steps proportional to the negative gradient of $L(x, y)$, thus *walking* towards the minimum of $L(x, y)$ [21]. The *walking* is performed by updating the weights of

the NN iteratively during the networks training.

$$f^* = argmin \sum_{x,y} Loss(f(x), y) \tag{2.1}$$

In supervised learning, a model is trained by giving a label to each data sample, the labels are simply the answers to what each data sample represent. The network is then able to process the difference between the labels of the data and its predictions such that its predictions converge towards the labels of the data [21]. If the model has been able to generalise the relationship between the training data and its labels then the model will be able to accurately predict the label of the previously unseen data.

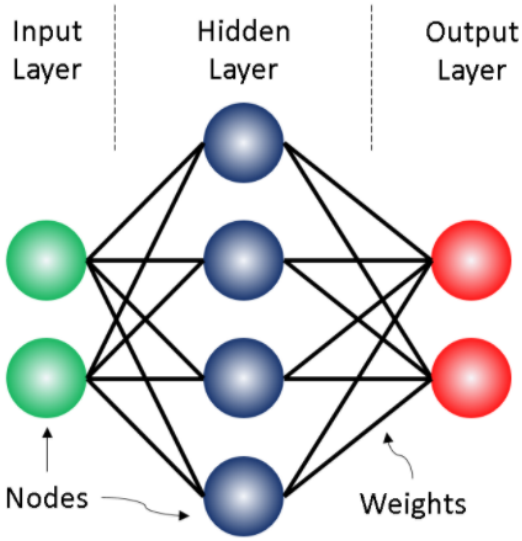


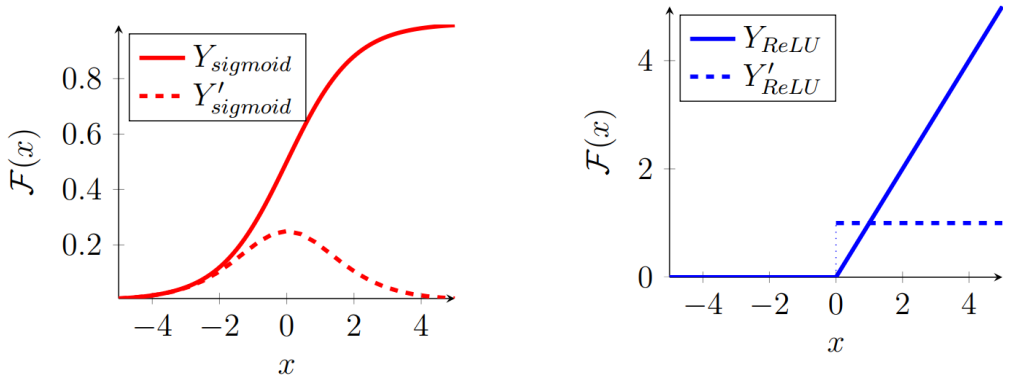
Figure 2.1: Example of a simple fully connected neural network.
Image source: mlnotebook.github.io [1]

Neural networks are a form of computational learning system that are used to solve complex mathematical tasks and interpret data [21]. The general structure of a NN consists of one input layer which receives the data to be processed, several hidden layers which process the data received in the input layer, and one output layer which makes the final processing and formats the output of the model, an example is given in Figure 2.1. Each layer generally consists of several *neurons* that each applies a nonlinear operation to their inputs. Examples of common operations are presented in Equation (2.2) and (2.3) which depict the Sigmoid and ReLU function respectively, in Figure 2.2 their characteristics are visualised. A neuron in a layer is generally only connected with the layer in front and behind it and does not interact with the neurons in the same layer as itself. The structure of a NN can thus be described in short as a series of non-linear operations interacting through linear connections.

$$Y_{\text{sigmoid}} = \mathcal{F}(y_{\text{in}}) = \frac{1}{1 + e^{-y_{\text{in}}}} \tag{2.2}$$

$$Y_{ReLU} = \mathcal{F}(y_{in}) = \begin{cases} 0 & \text{for } y_{in} < 0 \\ y_{in} & \text{for } y_{in} \geq 0 \end{cases} \tag{2.3}$$

The application of classifying data using a NN and the supervised learning approach is extremely broad in terms of possible field of application. This broad range is due to the requirement for application is that of data with corresponding labels for training the network. Classifiers trained using supervised learning can today be found in systems that handle facial recognition, medical imagery, voice recognition, and NLP among others.



(a) Plot of the range of the Sigmoid activation function and its derivative.

(b) Plot of the range of the ReLU activation function and its derivative.

Figure 2.2: Plot of the characteristic shape of the Sigmoid and ReLU activation functions and their derivatives.

2.2.1 Convolutional neural networks

One possible way of structuring the neurons in a NN and the connections between the layers is called a Convolutional neural networks. A CNN is distinguished by its use of convolutional layers which try to mirror the structure of the human visual cortex through stacking layers on top of each other that processes and identifies complex features [22]. An example of how the calculations are processed can be seen in Figure 2.3.

The workhorse in convolutional layers can be described as many small templates, generally called a kernels, which are processed over the input of the layer. This causes areas that match the pattern of the kernel to produce a large positive output. While areas of the input which match the kernel pattern poorly or not at all produce an output which trend towards zero [22]. An example of a one-dimensional kernel is visualised in Figure 2.4. By processing the inputs as a group, the features

that a convolutions layer learns can be considered as a form of correlation between different input combinations, and its output.

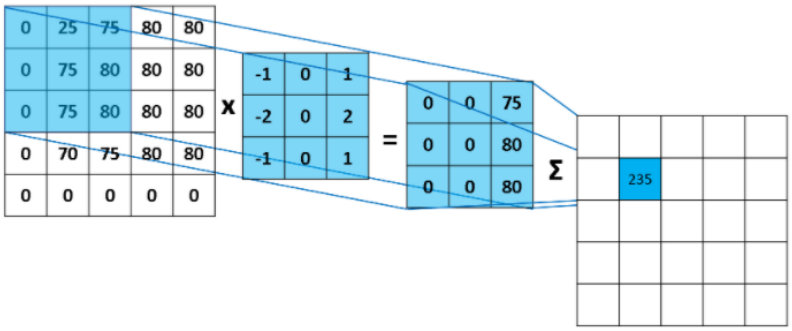


Figure 2.3: Example of a single calculation pass for a neuron in a convolutional layer using a 2-d kernel.
Image source: mlnotebook.github.io [1]

The use of kernels in a CNN is what distinguishes it the most from a fully connected feed-forward network. The kernels also make the number of trainable parameters in a CNN compared to a fully connected network far smaller given the same dimension of layers in the networks [22]. This is due to the parameters of the kernel being reused as it is processed over the input data of the layer. Thus is there no need to have a mapping between each neuron in two subsequent layers, which is the case for fully connected networks.

$$\begin{bmatrix} 1 & 0 & 1 \end{bmatrix}$$

Figure 2.4: Example of a 1d kernel in a CNN layer.

In conjunction with convolutional layers, a type of layer called pooling layer is often used. The pooling layer acts on the output of a convolutional layer by downsampling its feature map. There are two common approaches of how to perform the downsampling called max pooling and average pooling. The difference between the two methods is that max-pooling selects the feature with the highest value from the feature map while average pooling calculates the average of the map as the output. The size of the feature map which the pooling layer acts on is task-specific but can generally be seen as a small window that slides across the entirety of the layer's input. The pooling makes it possible to take into consideration small movements of the features of the input to the convolutional layer, thus making the CNN able to more robustly handle small variations of its inputs.

2.2.2 Integer representations

Before any neural network NLP model can make a prediction the input data needs to be translated from text to a representation that the computer can understand.

The representations task is to create a one to one mapping between each unique word w in the dictionary W of the input corpus to a unique token t, where $t \in T$ and thus $|W| = |T|$ i.e. the total number of words in the integer library is equal to the number of unique words in the corpus [23]. The number of unique words in a corpus often explodes as the size of the corpus increases, this makes it so that $|T|$ is often less than $|W|$. The limitation of T is frequently implemented by setting a threshold for the size of $|T|$ and only representing the $|T|$ most frequent words in the corpus. This limitation will cause documents in the corpus to only be partially represented, due to some words inevitably lacking representation in T.

2.2.3 Word embeddings

Word embeddings can be considered as a more complex representation of word than what can be done through a simple 1-d integer representation. The use of word embeddings gives a way to encode text to an efficient, dense representation in where words with a similar meaning will receive a similar encoding. The finalised word embeddings generally takes the form of an n-dimensional real-valued vector where each represented word has one unique vector connected to it. The embeddings are not directly specified by a developer but are instead a trainable parameter which is updated in the same way as a dense layer in a NN. Through this encoding, it is not only possible to capture similarities between words but is also possible to increase the dimension of the input to the model, compared to the single integer representation method presented in Section 2.2.2. This increase of dimensions creates a larger separation between the feature-weight combination which increases the probability that the data set is linearly separable [24]. Other representation methods such as one-hot encoding, where each represented word are represented by a sparse vector with a single high bit at the index in the vector corresponding to the words index in the dictionary T [25]. As the representation vector for each word will have the length of $|T|$, a sentence represented using one-hot encoding will be a combination of sparse vectors. Using word embeddings to represent one-hot encoded words will act as a dimensional decrease as long as the length of the embedding vector is smaller than the size of the token dictionary T, $n < |T|$. However, the vector will be transformed from sparse to dense, thus can it be seen as a more complex representation.

2.3 Data alteration using word filters

It is possible to apply a filter containing words that should receive special handling during the mapping process of words to integer representations [23]. Consider a word w included in the corpus dictionary W, and a dictionary G containing the unique words g of a filter. If $w \in G$ the mapping process from word to integer is stopped and an exception to the regular process is triggered for w. Examples of how the mapping process can be altered to handle the exception can be to never create a representation for the contained words in the filtered. A second method is to assign the same representation to all the words in the filter, thus masking the set of words. Another method is to exclude the entire sentence from receiving a representation. This method causes entire sentences to be removed from the corpus

but does not have the issue of potentially changing the meaning of sentences by partial representations. Pryzant's *et al* [26] research, regarding reducing subjective biases by suggesting edits to the training data using a word filter, suggest that their approach are able to provide useful suggestions to solve their task. Even though that their research were limited to the specific task of reducing subjective biases in the training data, their results could indicate that the method of selective filtering of the training data using a word filter could be a solution to the task at hand.

2.4 Methods for explaining neural networks

Neural networks have become popular due to their predictive power and flexibility in model fitting. Despite their power and popularity, there is some reluctance to fully adopt them. Neural networks are considered black-box models, meaning that the complex relationship between input features and the model's response cannot be easily described. However due to the use of the models in high-risk industries such as health care and finance, which are heavily regulated, the ability to interpret a model's behaviour has become a critical aspect of their application [27].

This section will present the underlying theory used to connect the prediction of a CNN-model back to its inputs. The core method stems from the field of computer vision but can be used for NLP by viewing a sentences embedding dimensions as the pixels of an image.

2.4.1 Pixel-wise decomposition

The general goal of Pixel-wise Decomposition is to produce a metric which gives a scale to the contribution of a single input x_i on the prediction $f(x)$ made by model a f. By making this metric signed it will be possible to see which inputs x_i contributed positively or negatively to the model's classification. To produce this metric, the model is assumed to have a real-valued output thresholded at zero. This assumption can be converted into a mapping $f : \mathbb{R}^V \rightarrow \mathbb{R}^1$ with $f(x) > 0$ denoting that a correlation between the input and output of the model is present. The prediction $f(x)$ can be decomposed as a sum composed of the terms of the single inputs, x_i, and the number of dimensions of the input x^d [5].

$$f(x) \approx \sum_{i=1}^{V} R_i \tag{2.4}$$

In Equation (2.4) $R_i > 0$ indicates that information is present at input x_i which supports the prediction of the model while $R_i < 0$ indicates that x_i contains information that contradicts the prediction. Pixel-wise decomposition does not require any image labelling, segmentation, or specific training schemes to be applied to a model. But the model needs to be trained for the method to be able to produce results that connect the input to the prediction [5].

2.4.2 Layer-wise relevance propagation

The entirety of Section 2.4.2 is paraphrased from the study "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation" written by Bach *et al* [5]. Bach *et al* were the first to present the concept of LRP, and the relevant theory from their paper is presented in this section.

LRP was proposed by Bach *et al* as a concept to achieve Pixel-wise Decomposition as described in Equation (2.4). LRP is described by Bach *et al* as a set of constraints, and that any solution that follows and satisfies the constraints can be considered to be a form of LRP [5]. To achieve Pixel-wise Decomposition the LRP method assumes the model can be split into separate layers of computation. In the context of neural networks, this separation of computation is considered to be the activation functions in the layers of the neural network.

The LRP method considers the first layer to be the input of the model and the predictions of a model f to be the last. All subsequent layers between the first and the last layer are modelled as a vector z at position l with dimensionality $V(l)$, as described by Equation (2.5).

$$z^l = \left(z_i^{(l)}\right)_{i=1}^{V(l)} \tag{2.5}$$

For LRP to work the relevance score $R_i^{(l+1)}$ must be available for each input dimension of the vector $z_i^{(l+1)}$. If this condition holds then $R_i^{(l+1)}$ can be propagated to the lower layer $R_i^{(l)}$ by following the connections between layer $l+1$ and l. The propagation between layers and the mentioned condition of availability of relevance scores for each input dimension causes the following Equation (2.6) to hold.

$$f(x) = \cdots = \sum_{i \in l+1} R_i^{(l+1)} = \sum_{i \in l} R_i^{(l)} = \cdots = \sum_i R_i^{(1)} \tag{2.6}$$

Thus Equation (2.6) describes that by iterating from the output to the input layer it is possible to identify how the input information x influenced the output prediction $f(x)$. With this information, it can be noted in Equation (2.7) that by propagating down to the input layer the relevance for the input layer is the sum decomposition from Equation (2.4).

$$f(x) = \sum_i R_i^{(1)} \approx \sum_{i=1}^{V} R_i \tag{2.7}$$

To further describe the relevance calculation during a backward pass of a neural network the example described in Figure 2.5 can be considered. This example does not have any direct connection to the neural network used in this thesis and only serves as an explanatory example for LRP.

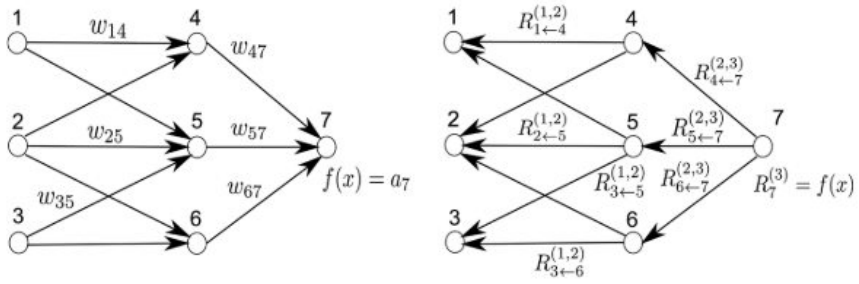


Figure 2.5: Visualisation of how the information can flow through a NN.
Left: Data flow in a neural network during prediction.
Right: Data flow during LRP propagation.

Image source: Bach *et al* [5]

In Figure 2.5 w_{ij} denotes the network weights, a_i the output of activation function from neuron i, for the network during classification. During the computation of the relevance scores for the neurons of the network, R_i^l describes the relevance score of neuron i at layer l, and $R_{i\leftarrow j}^{(l,l+1)}$ denotes the propagation from layer $l+1$ to layer l from neuron j onto neuron i. Figure 2.5 also shows how the relevance score of an arbitrary neuron is directly dependent on its local connections to the layer beside it. Since the output layer is the last layer of the model the relevance of the output layer is the prediction of the model, this connection is described in Equation (2.8).

$$f(x) = R_7^{(3)} = a_7 \tag{2.8}$$

To calculate the relevance scores for the layer directly after the output layer Equation (2.6) is applied with $f(x) = R_7^{(3)}$ as the initial value. For Equation (2.6) to hold Equations (2.9) and (2.10) must be true.

$$R_7^{(3)} = R_6^{(2)} + R_5^{(2)} + R_4^{(2)} \tag{2.9}$$

$$R_6^{(2)} + R_5^{(2)} + R_4^{(2)} = R_3^{(1)} + R_2^{(1)} + R_1^{(1)} \tag{2.10}$$

It can be noted from Figure 2.5 that the relevance score of any given neuron, except the output neuron, can be defined as the sum of all incoming relevance propagations. An example of this sum for neuron 2 is given by Equation (2.11).

$$R_2^1 = R_{2\leftarrow 4}^{(1,2)} + R_{2\leftarrow 5}^{(1,2)} + R_{2\leftarrow 6}^{(1,2)} = \sum_{k=4}^{6} R_{2\leftarrow k}^{(1,2)} \tag{2.11}$$

Equation (2.11) can be rewritten to a general case which is described by Equation (2.12), where i is the input of neuron k during prediction.

$$R_i^{(l)} = \sum_k R_{i\leftarrow k}^{(l,l+1)} \tag{2.12}$$

What needs to be noted is that the direction of the propagation during relevance computation is the opposite of the direction during prediction. The direction of the

local connections are reversed during relevance propagation and prediction, all inputs of a neuron during prediction becomes outputs during the relevance propagation. From Figure 2.5 it can also be noted that for Equation (2.6) to hold it is required that the relevance of a neuron is equal to the sum of its outgoing propagations. This condition is described by Equation (2.13).

$$R_5^{(2)} = R_{1\leftarrow 5}^{(1,2)} + R_{2\leftarrow 5}^{(1,2)} + R_{3\leftarrow 5}^{(1,2)} \tag{2.13}$$

A general notation for the condition described in (2.13) is presented in Equation (2.14), where i is the input to neuron k during prediction.

$$R_k^{(l+1)} = \sum_i R_{i\leftarrow k}^{(l,l+1)} \tag{2.14}$$

Equation (2.12) and (2.14) formulate in conjunction a final condition for LRP which is that the relevance which is propagated into a neuron must also be propagated out from it. If this condition does not hold then it would indicate that a relevance has either been created or lost in the model which would also cause Equation (2.6) to not hold.

With the knowledge of the above formulas and constraints, it is possible to write an explicit formula for LRP for the example in Figure 2.5. The LRP formula reflects the propagation from neuron to neuron during prediction by using the same reference propagation direction as during prediction and by incorporating the information that is transferred between the neurons.

The information neuron i propagates to neuron k is $a_i w_{ik}$, its activation function output a_i times the weight w_{ik} connecting the neurons. From this it is possible to reformulate Equation (2.13) such that the relevance score of neuron five is dependent on what is propagated to the neuron during prediction, the reformulation is described in Equation (2.15).

$$R_5^{(2)} = R_5^{(2)} \frac{a_1 w_{15}}{\sum_{i=1,2,3} a_i w_{i5}} + R_5^{(2)} \frac{a_2 w_{25}}{\sum_{i=1,2,3} a_i w_{i5}} + R_5^{(2)} \frac{a_3 w_{35}}{\sum_{i=1,2,3} a_i w_{i5}} \tag{2.15}$$

Thus finally a general equation describing how the relevance is propagated down through the network, which is based on how the information flows through the network during prediction time, be can presented in Equation (2.16).

$$R_{i-k}^{(l,l+1)} = R_k^{(l+1)} \frac{a_i w_{ik}}{\sum_h a_h w_{hk}} \tag{2.16}$$

From Equation (2.16) one can note that the equation is not defined when the denominator, $\sum_h a_h w_{hk}$, equals zero and diverges when it goes towards zero. A solution for these issues is presented in Section 2.4.3.

2.4.3 LRP epsilon rule

As mentioned in Section 2.4.2, as a comment to Equation (2.16), the relevance propagation can take on non defined values. This limitation of Equation (2.16) can

be solved by introducing a sign dependent balancing term ϵ in the denominator of the equation [5]. This introduction modifies Equation (2.16) to the form presented in Equation (2.17).

$$R_{i-k}^{(l,l+1)} = R_k^{(l+1)} \frac{a_i w_{ik}}{\sum_h a_h w_{hk} + \epsilon * sign(\sum_h a_h w_{hk})} \tag{2.17}$$

It can be noted in Equation (2.17) that ϵ is constrained by the condition $\epsilon > 0$ since breaking the condition causes the sign of $\epsilon * sign(\sum_h a_{hw_{hk}})$ to flip which can cause the local propagation to change sign or produce denominator equal to zero. Another property of Equation (2.17) is that the balancing term reduces impact of variance during the propagation by proportionally reducing small relevance contributions more than large [28]. The value of ϵ which is appropriate to use is dependent on both the model and its application. To avoid a degradation of the propagated relevance Binder *et al* recommends that ϵ is kept lower for deeper networks, due to ϵ being applied iteratively for each layer in the network [29].

2.5 Exploring corpus statistics

The data used to train and evaluate a neural network can contain valuable information regarding the network's predictions. The metadata can also be used to construct word filters which can be applied to act on specific attributes of the corpus. Two methods are presented in this section which extracts statistical information from a corpus regarding the distributions of words between the classes in the corpus.

2.5.1 Normalized comparative occurrence frequency

To investigate if some words are occurring more frequently in either of the available classes in a data set the Normalized comparative occurrence frequency (NCOF) can be calculated for each token. NCOF is similar to the method proposed by Rayson and Garside but is extended by giving a sign and a scale to the discrepancy between how often a word is occurring in the classes available in a corpus [30].

To calculate the NCOF for a data set S with samples s_i with targets $y(s_i) = k_i$, where $k_i \in K$, and K is a set containing all the targets of S and can be split into subsets equal to the number of target classes belonging to S, with each subset containing one unique target class. S is split and sorted based on the target for $s_i, \forall i \in I$ which will produce P, with $p \in P$, number of subsets each containing one target class. All data samples need to undergo some preprocessing which represents each word with a unique token, stemming and other preprocessing methods can also be performed, as long as the result produces a representation with a one-to-one connection between word and index.

The next step in calculating the NCOF is to summarise the number of occurrences of each unique token for each subset of the target classes into an array, o_j^p which contains the total number of occurrences for each index j of each token for the given subset p. The second to last set of calculations can then be performed by taking the

difference in occurrence frequency between a chosen subset p and the remainder of the subsets $P - p$ as described in Equation (2.18).

$$COF_j = o_j^p - \sum_{k,k\neq p}^{P} o_j^k \tag{2.18}$$

What can be observed from Equation (2.18) is that COF_j will be positive if the token at index j is more frequent in the subset p than in the remaining subsets $P - p$. It can also be noted that the COF is calculated through the perspective of o^p, meaning that if the COF is to be calculated for all subsets P then there will be an equal number of COF arrays as there are subsets, each with the perspective of one classification class.

The occurrence frequency of the indexes in each class can be normalised so that disparities between the number of instances of data samples belonging to each class does not skew the results. The normalisation of the COF results is performed by following Equation (2.19). Where the denominator of Equation (2.19) is equal to the total occurrences of all words in subset p.

$$t_j^p = \frac{o_j^p}{\sum_j o_j^p} \tag{2.19}$$

Which results in that the normalised COF_j, $(NCOF_j)$, is calculated through Equation (2.20)

$$NCOF_j = t_j^p - \sum_{k,k\neq p}^{P} t_j^k \tag{2.20}$$

By normalising the comparative metric t_j^p now describes the probability of a specific word occurring as an element in a sentence belonging to subset p. As this metric always will have the range of $[0, 1]$ the comparison of t_j between the classes of the data set, Equation (2.20), will give a more balanced result regarding how common words are in the respective classes of a data set.

2.5.2 Term frequency–inverse document frequency

Term frequency–inverse document frequency (TF-IDF) is a metric intended to reflect how important each unique word is in a corpus or collection of data [31]. The general approach of implementing TF-IDF for corpus analysis is as follows. For the set of documents D, consisting of individual document $d \in D$ each containing a set number of words. The TF-IDF score for each word w in a document d is calculated through Equation (2.21) [31].

$$w_d = f_{w,d} * \log\left(|D|/f_{w,D}\right) \tag{2.21}$$

$f_{w,d}$ represents the term frequency in TF-IDF, the number of occurrences word w has in document d [32]. $\text{Log}(|D|/f_{w,D})$ is the inverse document frequency (IDF), where $|D|$ is the number of documents in corpus D, and $f_{w,D}$ is the number of documents containing w in corpus D [33].

Given changing values of $f_{w,d}$, $|D|$, and $f_{w,D}$ a few different situations for the TF-IDF scores for words a document may occur. Given a situation where $|D| \approx f_{w,D}$, that a word w is present in almost every document in the corpus D [31]. This would entail that the contribution of the IDF part to w_d trend toward zero. If for a document in this situation $f_{w,d}$ for a given word is large, this would mean that $w_d < f_{w,d}$, since the IDF suppresses w_d. This would imply that the specific word w is common in both the corpus and the specific document. But could hold some importance in the context of the specific document. An example of words that could display this TF-IDF characteristic could be plaintiff or defendant in the context of legal documents.

The contrary example would be if $f_{w,d}$ is large while $f_{w,D}$ is small. $\text{Log}(|D|/f_{w,D})$ would then be quite large which would lead to w_d also becoming large. Of the two presented situations, this can be considered as the most interesting one. This is due to words with a high w_d are important in the context of d but not frequently occurring in D, thus implying that the word is interesting in a subset of the data [31].

3

Method

This chapter describes the steps taken in this study to apply the theory presented in Chapter 2 to go from raw data to the results presented in Chapter 4. Steps such as the preprocessing of the data, the creation of the word filters, and the model used for the classification are among other topics that will be introduced.

3.1 Workflow

To clarify the workflow used in this thesis the major steps are outlined in Figure 3.1 and 3.2. The NCOF and TF-IDF methods can be considered as data-focused, meaning that the filters are applied and used independently of the model at hand, while the LRP method is dependent on the model used to solve the task and thus requires a different approach for creating a word filter.

The process for the NCOF and TF-IDF filters starts with the preprocessing of the training data, how the preprocessing has been performed in this study is presented in Section 3.3.1. The next step is to perform the analysis of the metadata contained in the training data set. This step differs between the NCOF and TF-IDF approach, and the process for this step is outlined in Section3.4.2 and 3.4.3 respectively, related theory can be found in Sections2.5.1 and 2.5.2. When the metadata has been extracted the next step is to use that information to create the word filter which will be applied to the training data. What the term word filter entails and the intent of applying them is presented in Section3.4, the related theory for the application can be found in theory Section2.3. Applying the created word filter to the training data is the final step before retraining of the model takes place. The application of the filter is performed by using a process that will be named Sentence dropout. The details of this application process are explained in Section 3.5. When the application of the filter to the training data is completed the final two steps are the training and evaluation of the chosen NLP model. The structure of the chosen model is presented in Section 3.6. Section 3.7 presents how the evaluation of the classification result will be performed.

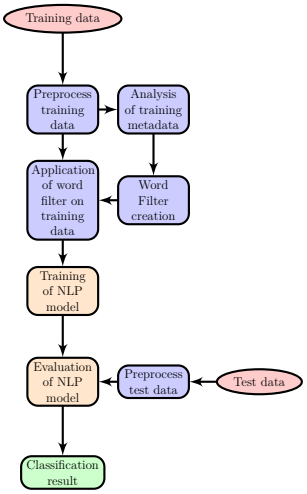


Figure 3.1: Visualisation of the workflow used for creating the NCOF and TF-IDF word filter.

The workflow for the creation of the LRP based filter differs from the NCOF and TF-IDF workflow in the sense that the process is a feedback loop. From Figure 3.2 it can be noted that the process of creating the LRP word filter cannot be performed before a trained model is present. After LRP is performed on the trained model and a relevance score is calculated for all inputs to the model, theory related for how the relevance score is calculated can be found in Section2.4.1. The workflow of creating the word filter, its application, and the training and evaluation of the results are the same for the LRP, NCOF, and TF-IDF methods.

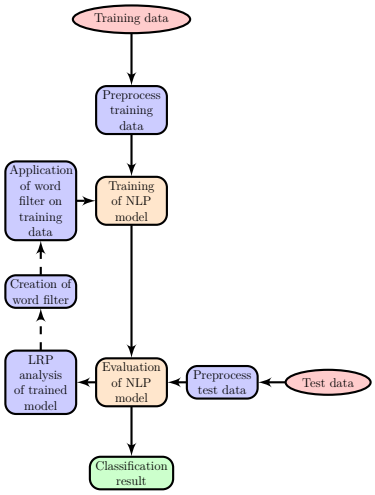


Figure 3.2: Visualisation of the workflow for creating and applying the LRP word filter. The whole line represents the workflow for the first iteration of training a model and creating an environment in which the LRP method can be performed. The dotted path takes priority first when the model has been fully trained.

3.2 Data set

The data used in this study is owned and produced by the company Recorded Future and is not in the public domain. The corpus is constructed for usage in a binary classification task and contains 63,536 sentences each associated with a binary label indicating if the sentence refers to a terror incident or not. Of these data samples, roughly 40,000 sentences have been manually labelled by both Recorded Future employees and third-party annotators. The remaining data samples have not gone through a manual inspection and are general data assumed to be negative and have received a negative label. The data set is a collection of sentences that handle the subject of terrorist incidents. The definition of a terrorist incident used by Recorded Future during the annotation process of the data set is defined as the following:

> "A terror incident can be defined as violence against civilians for political, ideological or religious reasons, as a means to create terror among people. It also includes violence directed towards infrastructural systems, like electrical grids or transportation systems." - Recorded Future

The relation between positive and negative label instances in the corpus is set to be roughly 19% positive and 81% negative. The source of the data is social media and various news sites.

3.3 Preprocessing

Before any NLP model can make predict the content of a sentence, the sentence needs to be translated from text to a representation possible for the computer to understand. The preprocessing steps taken in this study are presented in this section.

3.3.1 Corpus preprocessing

The first step of the corpus preprocessing will be to split the original data into separate subsets used for training and testing of the NLP model. The division between training and testing will be set to 66.7% and 33.3% of the total corpus size respectively. This division results in a training set composed of 42,569 sentences, of which 19% are instances of the positive class and the remaining 81% instances of the negative class. The test set consists of 20,967 sentences, the division between the positive and negative class is the same as in the training set.

The following steps will be performed on both the training and test data sets but to avoid repetition, the process will only be presented in the context of the training data set.

Each sample in the training data set was originally formatted as a list with one element containing the entirety of the sentence. Because of this, each sample will need to be split into a list containing each word of the sentence as separate elements, this process will be performed by splitting the sentences on whitespaces.

Because the sentences are to be split at whitespaces, the next step in the preprocessing is to remove punctuation marks and other characters that could be present beside words that are not whitespace-separated. This choice of preprocessing is due to the limitations of the integer representation of words, described in Section 2.2.2. As the method considers `foo` and `foo!` as two unique words the addition of punctuation marks could dilute the integer representation dictionary. This limitation also leads to the need of transforming all letters in the entire corpus into lowercase.

3.3.2 Integer representations of words

The default tokenizer available in the Keras library was chosen to transform each sentence in the training set to an integer representation [23]. The representation will be a sequence where each word in the sentence to be represented is exchanged for a corresponding integer. The structure of the integer representation indexation is such that the more common a word is in the context of the entire corpus the lower the integer representation it receives, such that the most common word in the corpus will be represented by the integer 1. Thus will the integer representation of a word and its index in the representation dictionary be the same. The representation dictionary will be constructed from the training data set exclusively, given that the information contained in the test set should be hidden from the model during training. By including the test data during the construction of the representation dictionary information regarding the content of the test data would be made available to the model prematurely.

3.3.3 Integer representation dictionary size

The size of the representation dictionary is limited to 5000 unique indexes. This limitation of the dictionary means that it will only be capable of representing 5000 words from the training set. Since the internal indexation of the dictionary is such that the more common words were awarded a lower integer representation. The dictionary size limitation of 5000 tokens means that only the 5000 most common words in the corpus were able to receive an integer representation.

The reason for limiting the number of representations to 5000 tokens is both to reduce the cost of running algorithms over the entire integer dictionary and due to Hapax legomenon. Hapax legomenon is an expression that describes words that occur only once in a document or a set of documents so that a robust interpretation of its meaning in the context of the document cannot be provided. As these words are assigned high integer representations due to their sparsity the integer dictionary size can be limited so that fewer hapax legomenon words receive an integer representation.

3.3.4 Word embeddings

In this study no pre-trained word embeddings will be used to represent the available data. Pre-trained word embeddings are defined as any public available embeddings

trained on data other than the data set available for this project. Instead new embeddings will be be created and trained using the available data set. The embeddings will be created and trained using the standard embedding layer available in Keras [23] with each word receiving a 50-dimensional vector representation.

3.3.5 Sentence padding

The model to be used in this study will use static sized input layer. This puts the requirement on the preprocessing to transform the data to this predetermined length. If the input data fails to meet this requirement the model will not be able to process it. The sentences will thus be padded such that all have a length of 75 elements. This specific length is chosen from the distribution of lengths of the sentences in the data set. As can be noted in the histogram presented in Figure 3.3 the majority of the sentences in the data set have a length of 50 words or lower. We used post-zero-padding meaning that sentences shorter than 75 will have the difference in length filled with zeros. The padding will be performed after the tokenization of the data set.

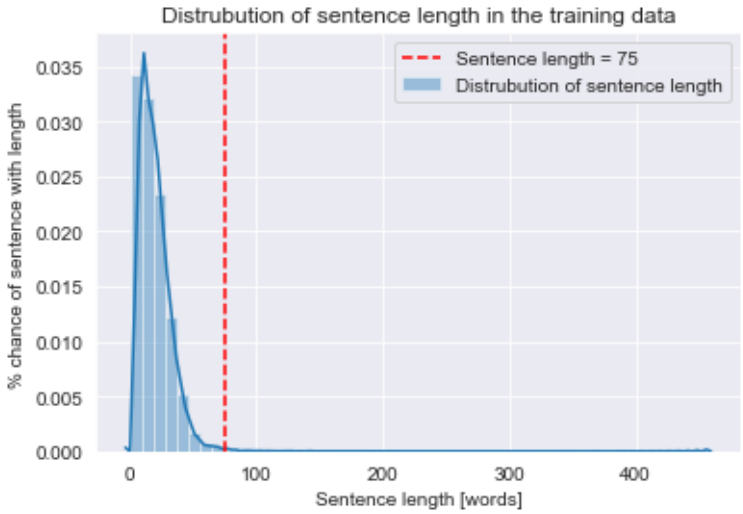


Figure 3.3: Histogram over the sentence length distribution in the training data set. The dotted line depicts the chosen maximum allowed sentence length in the training data.

The option to use a variable-sized input for the model is possible in the Keras framework [23]. The solution to use variable-sized inputs requires the data to be split and sorted into batches of similar length. These batches will then have to be padded such that the length within each batch is consistent. The more complex implementation of using a variable-sized input, together with the uncertainty during the design of the model of how LRP should be implemented when the input size is variable made using a fixed input size the best solution given the scope of the project.

3.4 Word filters

To investigate and evaluate if any of the methods proposed in Sections2.4.1, 2.5.1, and 2.5.2 can aid the identification of spurious correlations in a model, the words that the methods identify need to be filtered from the corpus. This will be done to see if the alteration has an impact in a model's prediction. This section presents the process of how to produce these word filters and how they are applied to the corpus.

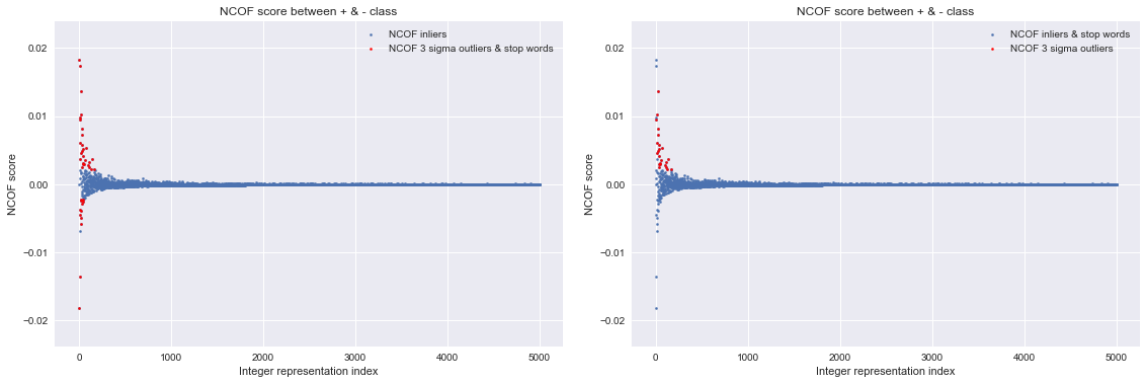
3.4.1 Selection of sigma for outlier identification

The selection of thresholds during the identification of outlier words and the creation of the word filters can be seen as a tunable hyperparameter of the methods. Given that any word that is identified as a statistical outlier within each separate method can be considered as interesting, regardless of used threshold, in the context of bias identification.

For this study, the threshold for the identification of statistical outliers will be set to $\mu \pm 3\sigma$ regardless of the method under investigation. This limitation is made so that the results of the application of each method will only show the most extreme cases of words with a skewed distribution in the data or weighting in the model. The selection of the $\mu \pm 3\sigma$ threshold will also entail a more concise and clear view of the results for each method. With a naive assumption that the results of each method will be normally distributed, each method will be expected to identify roughly 15 words as outliers from the dictionary of 5000 words, given that $\mu \pm 3\sigma$ would be expected to entail $99,97\%$ of the data. By shifting the threshold to $\mu \pm 2\sigma$ 5% of the 5000-word dictionary would now be expected to be identified as outliers, this represents 250 words, which would be impossible to present in an concise format.

3.4.2 NCOF based filter

By following the steps presented in Section 2.5.1 a score for all words contained in the dictionary of the integer representation will be produced. From this score the $\pm 3\sigma$ outliers from the scores mean, μ, will be selected. The words which will be included in the filter will thus be those which are the extreme outliers in regards to a skewed representation between the two classes in the data set. If any of the identified words are contained in the list of stop words presented by the NLTK Python framework they are removed from the filter [34]. There is no clear definition of what is considered a stop word but generally the most common words in a language are considered.



(a) NCOF score for training data

(b) NCOF score for training data, stop words removed from outliers

Figure 3.4: Results from NCOF calculation on the training data. 3σ outliers, used in the NCOF word filter, highlighted in red.

By comparing Figure 3.4a and 3.4b it can be noted that the stop words which were contained in the $\pm 3\sigma$ outliers mainly were skewed towards the negative class, and the removal of them causes the majority of the filter to consist of words more common in the positive class.

3.4.3 TF-IDF based filter

To create a filter based on TF-IDF we start with separating the data based on the associated label of the sentences. This is done so that TF-IDF scores can be produced for the positive and negative classes separately, where the TF-IDF score is calculated in accordance with the theory presented in Section2.5.2. From these scores, the $+3\sigma$ outliers will be selected as the base for further possessing.

Since the TF-IDF score is positive for a given data sample there is no negative score that can be taken into consideration in contrast to the NCOF filter method. Since this approach will produce two separate sets of outlier words, one from each class, the sets will need to be joined into one. This will be performed by taking the symmetric set difference between the $+3\sigma$ TF-IDF outliers from the positive and negative classes. The choice of using the symmetric set difference between the classes is made because this approach manages to create a combined set of words that have a high TF-IDF score in one class but not the other. Thus the words that are deemed important for each class will be selected for filtering, but the words that are important in both classes will be deselected. When the symmetric set difference between the positive and negative class is produced any remaining stop words will be removed from the set.

3.4.4 LRP based filter

In contrast to the two other methods for preparing a word filter the LRP based approach does not directly focus on the content of the training data but instead on what a model has learned from it. To produce an LRP filter a trained model is needed, the first step will thus be to train a model using the unaugmented training data. LRP using the epsilon rule, with $\epsilon = 0.01$, will then be applied to the trained model and a relevance score for each sentence in the training data is produced. The LRP score for each sentence will then be sorted into the confusion matrix quadrant which corresponds to the model prediction in relation to the label of each sentence. This step is performed so that the LRP score for all sentences which have been misclassified by the model during the final training epoch can be extracted. These misclassified sentences are interesting in the context of creating a word filter since these sentences contain words that the model possibly has overfitted on. This assumption is made due to the model has shown that it is not able to correctly classify the sentences, thus the sentences must contain some information that creates these erroneous classifications.

From the sorted LRP scores of the misclassified sentences, the $+3\sigma$ outliers are selected to be included in the filter. Only the positive LRP outliers will be selected since a positive relevance score is always in support of the model's classification, as explained in Section 2.4.1. This means that the positive LRP outliers for misclassified sentences represent the words that are deemed as most relevant for misclassified sentences.

3.4.5 Limitations of presented methods

A clear limitation of both presented frequency-based methods is that they are unable to process synonyms or inflection of words as a single entity. This can be noted in both Equation (2.18) and (2.21) by the sense that both equations are calculated for every unique word in d. To solve this issue a user would need to either preprocess the data or post-process the results of the methods to gather the results of the synonyms and inflections.

An issue that only appeared when creating and working with the LRP based filter was the presence of the embedding layer in the model. Since the LRP methods require predefined knowledge of all the connections in a model to be able to perform a backward pass on the entirety of the model, it was not clear how the relevance should be propagated from the input layer of the CNN model to and through the embedding layer. The solution for this was to separate the embedding layer from the CNN model such that the relevance could be propagated through the CNN model and embedding layer individually.

3.5 Application of word filters to data

The suggested method will only be applied to the training data and never to the test data. This distinction is made due to the premise of never disclosing any information contained in the test data to the model. Filtering the test data can be considered as modifying the evaluation of the model to better match the sought after result, which will make the test non-representative for the population the unfiltered data is trying to model.

The method used to apply the three different filters to the training date will be the method of sentence dropout. This method operates on the data set by giving a sentence a probability of being removed from the corpus if it contains a word that is present in a filter, such that if a word w_i is contained in both the sentence s_j and the filter w_f, $w_i \in s_j$ & $w_i \in w_f$, there is a probability of removing the entire sentence s_j. Since the possibility exists that a sentence can contain more than one instance of $w_i \in w_f$ the relative probability of a sentence being dropped from the data set is higher than what the dropout rate is set to. Therefore, the specific value of the dropout ratio is less important than the actual number of sentences removed when comparing the different methods of producing the word filters.

This selection of filter application method and regularization method was chosen since the filter creation processes mainly focus on the skewed presence and distribution of specific words, and not the the meaning of the words in their sentences. The chosen method regularizes the data without altering the sentences, so the meaning of the remaining sentences is preserved after the regularization process.

3.6 Model structure

The model used to solve the binary classification problem presented by the data set will be a CNN model. The model will be constructed and evaluated using the Python framework Keras [23].

The structure of the model used in this study is presented in the following list. This structure resulted in a relatively lightweight model with $350,751$ trainable parameters which means that training the model and evaluating LRP for any given data set is comparatively cheap. The trainable parameter count is important to keep low for this study since the computational cost of performing LRP is in direct relation to the number of parameters in the model it is performed on, as mentioned in section 2.4.1, and the number of parameters often explodes when the complexity of the model increases.

- Input : Integer sequence of length 75
- Layer 1: Embedding layer: 50 embedding dimensions
- Layer 2: Dropout layer: drop rate = 0.2
- Layer 3: 1D convolutional layer: 250 filters, kernel size: 10

- Layer 4: Max pooling layer
- Layer 5: Fully connected layer: 250 neurons, ReLU activation function
- Layer 6: Dropout layer: drop rate = 0.2
- Layer 5: Fully connected layer: 1 neuron, Sigmoid activation function

3.7 Evaluation of word filter application

The evaluation of the three different word filters will be conducted through iterative evaluation of each available filter applied to the static training data. The iterative evaluation will be performed by training and calculating the test accuracy, defined in Equation (3.1), for the model, given one of the filters, for a set number of times, common to all filters. In Equation (3.1) TP is the number of sentences classified as true positives, TN true negatives, FP false positives, and FN false negatives, following the quadrants in Table 3.1.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

		Data label	
		Positive	Negative
Model prediction	Positive	TP	FP
	Negative	FN	TN

Table 3.1: Example of a binary confusion matrix describing the four confusion quadrants.

Sentence dropout will be applied to the data in three different levels of aggressiveness. The three levels will mean that the number of available data samples for the model to train on will decrease as the filter application aggressiveness increases. For each level of filter, application the test accuracy and classification result will be evaluated.

The number of remaining training data after a filter has been applied to the training data will be set as roughly 39, 37, and 33 thousand data samples. For the tests of filter application to be comparative to a filter-free model a baseline will be used. To emulate the filter application on the baseline model randomised training data will be removed such that the remaining data match the three different levels of filter application.

To evaluate how the filter application affects the weighting of the model's inputs, LRP will be performed on the model before and after the application of the filters. By performing LRP before and after the application of the three filters it will be possible to calculate and identify the differences in the weighting of the model's inputs before and after filter application. From the LRP scores the 3σ outliers in regards to which words that drive the model towards its classification result will be

selected and a comparison between before and after the application of the filters will be conducted.

This experiment will hopefully show that using the extracted information from the model or training data to regularize the data using a word filter does not drastically alter the population modelled by the training data set, thus indicating if removing overrepresented terms from the training data can be used to help a model generalise by altering the modelled population.

3.8 Identifying a spurious correlation

The first part of this experiment will be to make sure that there exist a known bias in the data set so that the methods ability for identifying it can be evaluated. The inducing of the bias will be performed by selecting one or several keywords present in the training data that would compromise the result of the model's classification. The keywords that will be used in this study are *mosque* and *mosques*. To induce a bias in the model from the training data connected to the keywords all sentences which contain either of the keywords will have its label changed to positive. This will hopefully, in the context of this test, induce a bias and the model will learn the correlation between the keywords and the positive class.

After the training data has been augmented and a baseline model has been trained the next step will be to produce three new word filters: one from the newly trained model, LRP, and two from the augmented training data, NCOF, and TF-IDF. From the filters, it will be possible to evaluate which words each filter method have deemed as interesting and thus evaluate if the spurious correlation picked up by the filters.

To investigate if applying the filters on the training data does reduce the influence of the bias on the model's classification, three models, each trained using data with one of the filters applied to them, will be created. The filters will be applied to the training data such that roughly 33 thousand sentences remain after the application, corresponding to the most aggressive application in previous tests. A baseline model trained using unfiltered data but with a reduced number of data samples will also be used for comparison in this test. Three example texts from the test data will be used for the evaluation of the filter application. The texts are presented below in the same formats as they are used as an input for the CNN model, not as they are formatted in the data set.

> Example text 1, label 0:
> "new zealand prime minister calls christchurch mosque attack a terrorist attack"
>
> Example text 2, label 0:
> "chinese muslims gather before eid prayers the end of the holy month of ramadan at the historic mosque on june 16 2018 in beijing china"

Example text 3, label 1:
"an assailant set fire to ibrahim mosque in hamilton ontario canada"

The evaluation of the effect of the filter applications will start with isolating all sentences in the test data which contain any of the keywords. It can be noted that the labels for the isolated sentences will be kept unchanged so that the test data contains sentences with the keywords either *one* or *zero*. The next step is to produce an LRP score for all models in regards to the isolated sentences and investigate which words have received the highest positive score for sentences classified as true and false positive. From the words with the highest LRP score, it will be possible to investigate which data the model deems as most important when making positive classifications and if the keywords are present among these words.

4

Results

This chapter presents the results which have been produced during the filter creation processes, identification of a spurious correlation, and the effects of applying a word filter to a model.

4.1 Result of filter creation

In the following sections, the results of applying the three methods for identifying interesting words in the context of the training data are presented. The methods used for the filter creation are described in Section 3.4.2, 3.4.3, and 3.4.4.

The results will be presented in both a visualisation, such that the class origin of the identified words is shown, and in plain text. It will thus be possible to observe how the contents of the word filters are extracted from the data and to identify the specific words which have been identified.

4.1.1 NCOF

It can be noted in Figure 4.1 and 3.4 that the majority of the integer index representation are balanced between the two classes. This can be noted as most indexes are grouped around an NCOF score of 0. It can also be noted that it is the index representations that are frequent in the corpus that have skewed NCOF scores to either of the classes. This insight can be made due to the majority of the skewed indexes are low valued which corresponds to frequently used words in the corpus, as described in Section 3.3.3.

By remembering the note from Section 3.4.2 regarding that the majority of the removed stop words were contained in the negative class and investigating Figure 4.1 it can be noted that the words which will be included in the finalised word filter are only from the positive class.

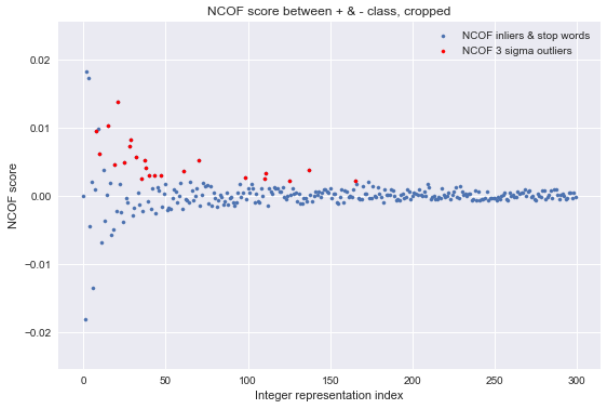


Figure 4.1: Zoomed version of Figure 3.4b, depicting the NCOF score for the first 300 integer representations. The 3σ outliers selected for the NCOF word filter highlighted in red.

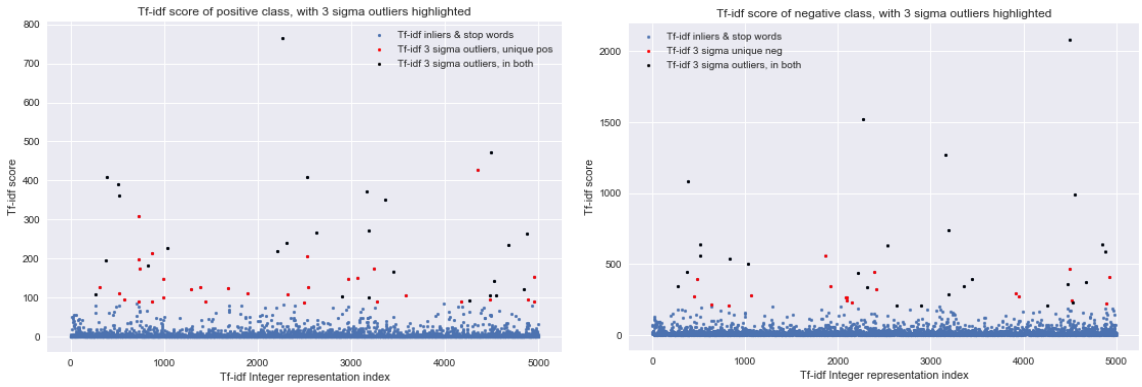
The results regarding that the finalised NCOF word filter only consisting of words which originates from sentences with a positive label are a bit unexpected. A point of interest can be noted from the comparison of Figure 3.4a and Figure 4.1 is that many of the stop words used in this study are are more common in the negative class than the positive. Since the Stop words are the 100 most common words in the English language it would be expected that they would be used in roughly the same frequency between the classes. Given that the NCOF score is normalised to take the distribution difference between the classes into account this disparity of the stop words frequency could indicate that there is difference between the language used in the positive and negative class.

Positive 3σ **outliers**			**Negative** 3σ **outliers**
attack	killed	wounded	
bomb	killing	wounding	
bomber	kills		
bombing	least		
car	near		
city	others		
detonated	people		
exploded	suicide		
injured	three		
injuring	two		

Table 4.1: 3σ NCOF outliers words presented in Figure 4.1, converted from integer representations and separated by their class origin. Alphabetically sorted.

4.1.2 TF-IDF

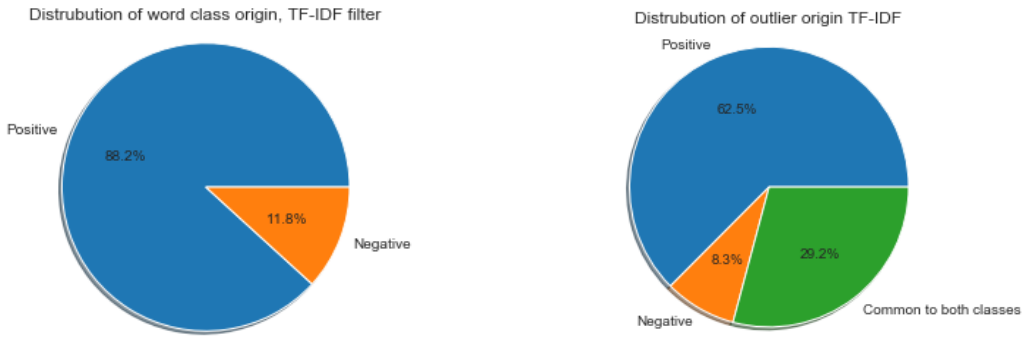
The resulting TF-IDF score for the integer representation of all words in the training data can be viewed in Figure 4.2. It can be noted from Figure 4.2 that the outliers which are common to both the positive and negative class generally receive a higher score than the words that are only present in outliers one of the classes. It can also be noted that there is a large overlap between the classes in terms of tokens with TF-IDF scores high enough to be 3σ outliers.



(a) TF-IDF score positive class **(b)** TF-IDF score negative class

Figure 4.2: Result of class separated TF-IDF score of the training data set. The 3σ outliers used in the TF-IDF filter highlighted in red. 3σ outliers which occur in both the positive and negative class are marked black.

By taking the symmetric set difference of the two set of outliers, as mentioned in Section 2.5.2, the words that are included in the word filter based on TF-IDF are presented in Table 4.2. The words are presented with their class origin. In total, the method identifies 30 words that have received a high TF-IDF score in the positive class but not in the negative, and 4 words with the same property in the negative class. This results in a filter that consists 88.24% of words contributed from the positive class and 11.76% from the negative class. These distributions can be viewed, together with the percentage of identified outliers that occurred in both the positive and negative class, in Figure 4.3.



(a) Class origin for words used in the TF-IDF filter

(b) Class origin for all 3σ TF-IDF outliers identified

Figure 4.3: Distribution of class origin of identified TF-IDF outliers, both for the subset used in the TF-IDF filter and for all identified outliers.

3σ **outliers from + class**			3σ **outliers from - class**
al	civilians	mosque	bus
attacked	dead	near	com
baghdad	detonated	others	new
bomb	district	pakistan	said
bomber	exploded	province	scans
bombers	four	security	
bombing	injuring	soldiers	
capital	kabul	suicide	
car	killing	terrorists	
city	kills	wounded	
		wounding	

Table 4.2: 3σ TF-IDF outliers words presented in Figure 4.2, converted from integer representations and separated by their class origin. Alphabetically sorted.

4.1.3 LRP

The resulting LRP scores for the misclassified training data, false positives, and false negatives, can be viewed in Figure 4.4 and Figure 4.5. Before the false positive and false negative LRP outliers are merged into the final LRP filter stop will be removed, the stop words are the same as those used in the two previous filters.

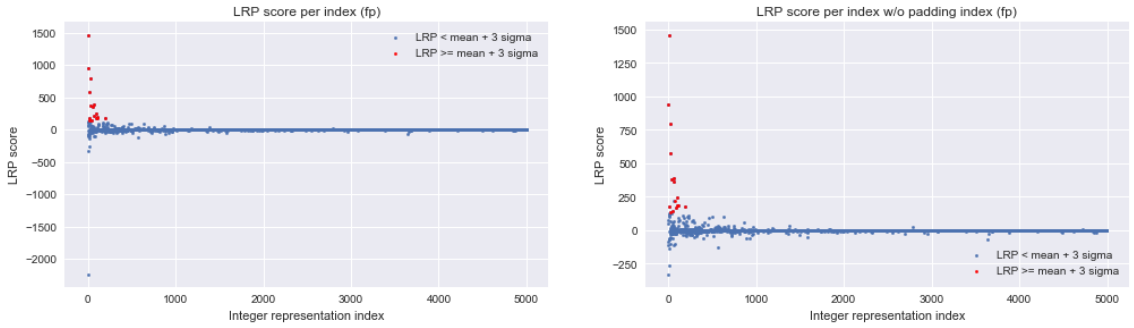


Figure 4.4: Summarised LRP score for each integer representation, score summarised for all sentences misclassified as false positive during the training of the model.

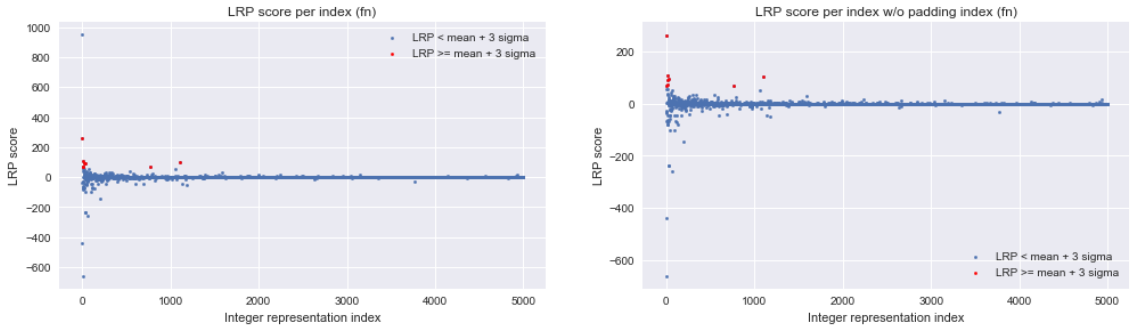


Figure 4.5: Summarised LRP score for each integer representation, score summarised for all sentences misclassified as false Negative during the training of the model.

The final words that were identified and applied in the LRP based word filter can be viewed in Table 4.3. The LRP method identified 14 outlier words in the context of their relevance score for sentences misclassified as false positive and only 2 words from sentences misclassified as false negative. This resulted in the LRP filter having an 87.5%, 12.5% distribution between contributions from false positive and false negative, the distribution is visualised in Figure 4.6.

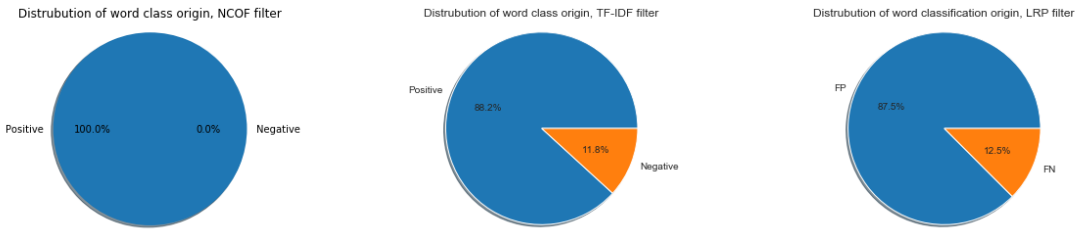
3σ outliers from FP		3σ outliers from FN
attack	exploded	2012
attacked	explosion	broke
attacks	killing	
bomb	opened	
bomber	suicide	
bombing	terrorists	
detonated	two	

Table 4.3: 3σ LRP outliers, translated from index representations back to the corresponding words used for the LRP word filter. Words separated by their confusion quadrant origin. Alphabetically sorted.

From Table 4.3 it can be noted that the words that are identified from the sentences classified as false positive can naively be called typical for the subject that positive class is about, given that the subject of terrorist incidents can be identified from the words. It can also be noted that the large overlap between these words and those identified as interesting from the positive class in the NCOF and TF-IDF methods points towards a connection between some of the identified words and the positive class. However, the outlier words identified from false positive classifications must all stem from sentences with a negative label. This could indicate that there are sentences with a language that is similar to that used the positive class, but have received a negative label. The reason for this could be either incorrect annotations, that the annotator of the sentence have misclassified it, or that the definition of terror incidents, presented in Section 3.2, allows sentences to talk about the subject very closely without receiving a positive label.

4.1.4 Comparison of origin distribution for filter content

From Figure 4.6 it can be noted that the majority contribution to the filters originates from positive class, or classification as positive for the LRP method. These results indicate that there are more words in the positive class that can be deemed as interesting in terms of occurring more exclusively in the positive class or managing to push the model towards a positive classification.



(a) **(b)** **(c)**

Figure 4.6: Distribution of class origin for words used in the three evaluated word filters.

4.1.5 Observations common to the three filters

It can be noted from the three Tables 4.1, 4.2, and 4.3 that the words used in the filters are unrepresented by words which originates from the positive class.

From the Tables 4.1, 4.2, and 4.3 it be noted that there is an overlap between all three methods in terms of which words are identified as interesting and selected for filtering.

4.2 Result of filter application

This section present how the class distribution of the training data set is altered by the application of the three different word filters presented in Section 4.1, it also presents how the accuracy of the model specified in Section 3.6 is affected by the filter application.

4.2.1 Implication of filter application on training data

When we apply a filter to the training data, the data is changed regardless of which filter is applied. How the filter is applied and the aggressiveness of the application will have a direct effect regarding the resulting transformation of the data.

By filtering the training by dropping entire sentences based on the content of the filter it is not only possible to change the content of the training set but also the distribution between the positive and negative class. As can be noted in Figure 4.7 the distribution between the positive and negative class was changed when the application method of sentence dropout was tested. As expected the percentage of positive sentences in the training data decreases as the aggressiveness of filter application increases, i.e. more sentences are removed. This behaviour is expected and occurs for all the tested filters due to them consisting of mainly words associated with the positive class. Thus sentences are removed from the positive class at a higher rate than from the negative class, thereby changing the composition of the training data set.

The change of distributions of positive instances in the training data that occurs when we apply the word filters is a direct effect of the chosen application method. If another application method would have been chosen, dropping words from sentences instead of entire sentences, for example, this behaviour would not be seen in the results.

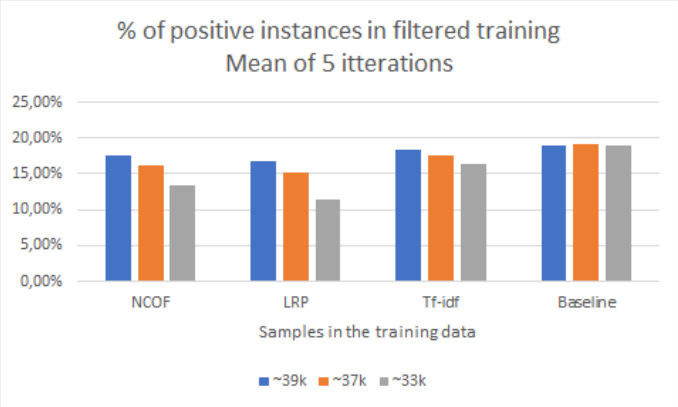


Figure 4.7: Percentage of positive samples in the training data after sentence dropout have been applied using the specified word filters at three different levels of aggressiveness.

From Figure 4.7 it can be noted that the application of the LRP filter has caused the largest decrease in positive instances of training data, from the unaltered 19% down to 11.5% for the most aggressive filter application. The negligible change in the percentage of positive instances for the baseline model is expected because the removal of data for this test case was randomised.

4.2.2 Impact on model accuracy

Figure 4.8 shows the test accuracy for the model for the three different aggressiveness of sentence dropout using the different filters. The results appear to indicate that the general trend for the test accuracy is that it decreases when the aggressiveness of filter application increases. The test accuracy for the filtered models can be noted as lower than the test accuracy of the baseline model given roughly the same amount of training data.

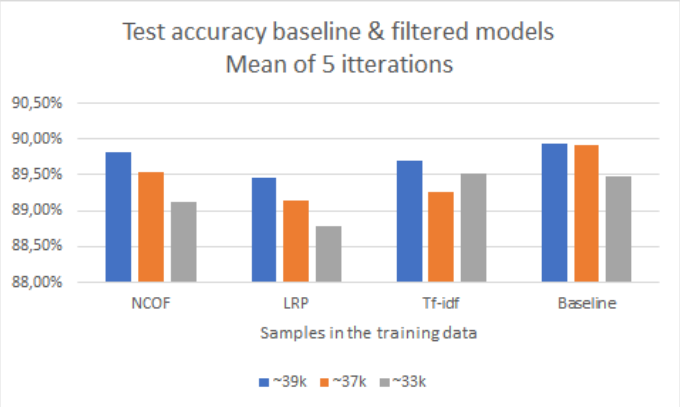


Figure 4.8: Mean test accuracy for the model with sentence dropout applied using the specified word filters at three different levels of aggressiveness.

4.2.3 Relevance score

Figure 4.9 visualizes the impact of each applied filter with regards to how the input words have affected the model's classification. From this figure, it can be noted that the weighting of specific input words is different compared to both the baseline model and the baseline model trained with a reduced data set. The application of the NCOF filter to the training data augments the data set to the degree that the model's classification of the sentence is changed. Thus it can be noted from Figure 4.9 that by applying the filter it is possible to alter how the model weights the input words during the classification of a sentence.

	new	zealand	prime	minister	calls	christchurch	mosque	attack	a	terrorist	attack	Predicted
Baseline	-0,23	-0,05	0,34	0,03	-0,47	0,08	0,01	0,00	-0,21	0,22	-0,05	0
Baseline (Reduced data)	0,00	0,00	0,03	-0,01	-0,04	0,03	0,03	-0,10	-0,20	0,14	0,00	0
NCOF	0,00	-0,01	-0,07	-0,01	0,02	0,16	0,00	0,17	0,15	0,15	0,07	1
LRP	0,000	0,001	-0,001	0,006	0,015	-0,002	-0,001	0,000	-0,005	-0,002	0,001	0
TFIDF	0,07	0,66	-0,04	-0,04	0,29	0,00	0,01	-0,08	-0,18	-0,16	-0,32	0

Figure 4.9: Relevance score for example sentence 1 for models trained on the unaugmented data set with the specified word filter. A positive score (red) is always in support of the model's classification.

4.3 Identifying a spurious correlation

The following results have been produced to meet the goals of the study as presented in Section 1.2. The method which has been followed during the experiment is outlined in Section 3.8.

The word filters that were created from the augmented training data set are visualised in Tables 4.4, 4.5, and 4.6. There are no major differences in the NCOF and TF-IDF filter between which words the filters have found to be of interest in the augmented training data set compared to the unaugmented set. It can be noted that the TF-IDF filter also managed to identify one of the keywords in the unaugmented data, as can be seen in Table 4.2.

Positive 3σ outliers		**Negative 3σ outliers**
attack	killing	
bomb	kills	
bomber	least	
bombing	mosque	
car	near	
city	others	
detonated	people	
exploded	suicide	
injured	three	
injuring	two	
killed	wounded	
	wounding	

Table 4.4: NCOF outliers from the training data, separated by their class origin. Outliers produced from the training data set with *mosque*-augmented labels. Alphabetically sorted.

3σ outliers from + class			**3σ outliers from - class**
al	civilians	mosque	bus
attacked	dead	near	com
baghdad	detonated	others	said
bomb	district	pakistan	scans
bomber	exploded	province	
bombers	four	soldiers	
bombing	injuring	suicide	
capital	kabul	terrorists	
car	killing	wounded	
city	kills	wounding	

Table 4.5: TF-IDF outliers from the training data, separated by their class origin. Outliers produced from the training data set with *mosque*-augmented labels. Alphabetically sorted.

The filter created from the LRP method found more outlier words, from both false positives and negatives, using the augmented training data compared to the unaugmented. A total of 33 words were found, of which 26 originated from words classified as false positive and 7 words from false negative sentences. The filter did not identify any of the keywords as interesting in the selection of misclassified sentences when evaluating the entirety of the training data set. The method did however identify both keywords as important words in sentences classified as true positive, which can be viewed in Table 4.7. The keyword will however not be included in the finalised LRP filter for the *mosque*-augmented data set due to the LRP filter method having been specified as only using words identified from sentences classified as false positive or negative.

By comparing the 3σ true and false positive outliers found using the LRP method, it can be noted that there is a large overlap between the outliers.

3σ **outliers from FP**			3σ **outliers from FN**
attack	detonated	people	2017
attacked	exploded	state	clashes
attacks	explosion	suicide	forces
bomb	explosive	taliban	operation
bomber	grenade	terrorists	police
bombers	injured	wounded	roadside
bombing	injuring		students
car	islamic		
civilians	killed		
dead	least		

Table 4.6: LRP outliers from the training data, separated by their confusion quadrant origin. Outliers produced from the training data set with *mosque*-augmented labels. Alphabetically sorted.

3σ **outliers from TP**					
alshabaab	bombers	civilians	killed	people	wounded
attack	bombing	dead	least	suicide	
attacked	bombings	detonated	mosque	taliban	
bomb	bombs	exploded	mosques	terrorist	
bomber	car	islamic	opened	terrorists	

Table 4.7: LRP outlier from the training data, outliers originate from the true positive confusion quadrant. Alphabetically sorted.

Table 4.8 presents the relevance score for the keyword *mosque* from the evaluation of example sentence 1. The results in Table 4.8a is produced from models trained on the unaugmented data set or the unaugmented data set with corresponding word filter applied to them. In Table 4.8b the process for the results is the same but the data set used for the training is the *mosque*-augmented data set. It can be noted from a comparison of the two tables that the *mosque*-augmentation indeed does drive up the weighting of the specific keyword in the baseline models. It can also be noted that the augmentation causes the baseline model to misclassify the sentence as a false positive. From Table 4.8b the application of the filters appears to lower the models weighting on the specific keyword since the relevance score for the keyword has been lowered.

Further results from the application of both the data augmentation and the filter application can be viewed in Tables 4.9a and 4.9b. The tables shows the relevance score for the word *mosque* from example text 2 and 3 evaluated on the models named in the table which have been trained on the augmented data. In these tables, it can once again be noted that the relevance for the keyword has been significantly lowered for the models which have had the word filter applied to them compared to the

baseline. From the evaluation of example text 3, it can be noted that the baseline model manages to correctly classify the sentence as a positive and the application of the filters managed to flip the classification to negative. In all three evaluations of the example texts, the model trained with the LRP augmented data manages to significantly lower the weighting of the keyword despite the filter not containing any of the specific keywords. This can be viewed in Table 4.8b, 4.9a, and 4.9b.

	Mosque	Predicted
Baseline	0,01	0
Baseline (reduced data)	0,03	0
NCOF	0,00	1
LRP	-0,001	0
TF-IDF	0,01	0

(a) Relevance score for models trained with unaugmented data

	Mosque	Predicted
Baseline	0,986	1
Baseline (reduced data)	0,827	1
NCOF	0,001	0
LRP	0,001	0
TF-IDF	-0,004	0

(b) Relevance score for models trained with mosque augmented data

Table 4.8: Relevance score for the word *mosque* for example text 1, label 0. Score calculated for models trained with the specified word filter and with or without *mosque*-augmented data.

	Mosque	Predicted
Baseline	1,459	1
Baseline (reduced data)	1,755	1
NCOF	0,000	0
LRP	0,001	0
TF-IDF	0,000	0

(a) Relevance score for example text 2, label 0, for models trained with *mosque*-augmented data

	Mosque	Predicted
Baseline	1,294	1
Baseline (reduced data)	1,142	1
NCOF	0,000	0
LRP	0,001	0
TF-IDF	0,000	0

(b) Relevance score on example text 3, label 1, for models trained with mosque augmented data

Table 4.9: Relevance score for the word *mosque* for example text 2 & 3. Score calculated for models trained with the specified word filter with the *mosque*-augmented data.

In Table 4.10 the tested models accuracy and error rate for the subset of the test data which contains the words *mosque* or *mosques* are presented. From the table, it is clear that the accuracy for the subset decreases for the model when a filter is applied to the training data. However, it must be noted that the evaluated subset consists of 372 samples with a distribution between the classes of 57.4 and 42.6% for the positive and negative class respectively. This distribution together with the accuracy results from the two baseline models indicates that the models consistently classified every sentence in the subset as positive, which were confirmed from closer inspection of the classification result. The accuracy of the three models which had word filters applied to them do lie close to the distribution of the negative class in the test data. The predictions of the three models do however not show a strong systemic error of unambiguously choosing a single class for their prediction. The models do show a tendency of favouring the negative class for their prediction but not to the extent that the baseline models exclusively select the positive class for all their predictions.

	Baseline	Baseline (reduced data)	NCOF	TF-IDF	LRP
Errors [#]	154	154	203	214	210
Accuracy [%]	58.6	58.6	45.4	42.4	43.5

Table 4.10: Error rate and % accuracy for the specified models on the subset of the test data that contain sentences with any of the two specified keywords.

By investigating Figure 4.10 and 4.11, which shows the 3σ outlier words from sentences classified as true or false positive. These words are those which have received the highest cumulative relevance in their confusion quadrant from the isolated test sentences containing either of the keywords. Both the baseline models have the two keywords as their only positive 3σ outliers for true and false positive classified sentences while the models with filter applied have neither.

A point of interest in Figure 4.10 is that the words *pakistan* and *pakistani* are included as outliers for the NCOF filtered model. These words could be considered as a protected attribute as it is referring to an ethnicity and the country associated with it.

Baseline	mosque	mosques								
Baseline (reduced data)	mosque	mosques								
NCOF	and	at	least	killing	dead	injuring	pakistan	kills	pakistani	update
LRP	a	and	were	while	children	resulted				
TF-IDF	at	exploded	kills							

Figure 4.10: 3σ LRP outliers from the training data for the specified models. LRP scores pooled from sentences classified as true positive.

Baseline	mosque	mosques								
Baseline (reduced data)	mosque	mosques								
NCOF	a	at	was	least	killing	his	following	earlier	response	condemned
LRP	a	was	sunday							
TF-IDF	in	at	that	three	city	20	years	march		

Figure 4.11: 3σ LRP outliers from the training data for the specified models. LRP score pooled from sentences classified as false positive.

5
Discussion

This chapter discusses the results presented in Chapter 4. The chapter begins by discussing the results and implications of filter creation, followed by a discussion of the results of identification of spurious correlations and how the application of the filter has altered the training data. The final sections of this chapter discuss the burden of responsibility of bias identification and removal, how the goals and aims of the thesis have been met, and the ethical implication of this thesis.

5.1 Filter creation

It was noted in Section 4.1 that the positive class was over-represented in all three identification methods in terms of the number of identified words. A reason for the under-representation of the negative class can be explained by the sentences in the negative class being too general for specific words to stand out as outliers for the subset of the data. Another reason for the disparity of words from the negative class could be that the negative class is roughly twice as large as the positive class.

A third reason for the large distribution difference between the classes could be that the positive class contains the data which can be considered as interesting for the binary classification task, since the sentences in the positive class are about the same general subject. It can thus be expected that the language will be somewhat coherent between the texts. In comparison, in the negative class where the sentences can be about any subject as long it is not the positive class. Thus it is reasonable that there will be a larger contribution of words from the positive class to the filter. The words that are specific for the positive class cannot be expected to be frequently occurring in the negative class, due to the difference in subject. And the words that are frequently occurring specifically to the negative class are stop words, given the general subject matter of the negative class. This behaviour, noted in Figure 3.4, can be considered as a reasonable explanation for the lack of identifying words for the negative class.

The biggest limitation of the three identification methods is their inability to process inflections of the identified words, as mentioned in Section 3.4.5. The implication of this can be viewed in Tables 4.1, 4.2, and 4.3 where multiple words with the same root are identified, for example the identified permutations of suffixes for the root *bomb*. It is a possibility that there are more inflections of *bomb* which are outliers in the identification methods but do not appear in the results due to the 3σ thresh-

old. This issue could be solved by implementing the ability for the filter to process word stems or lemma, such that the results are summarised for each word stem. Off the shelf solutions for this problem are available in many popular NLP python frameworks. This solution to the problem has not been tested in this thesis and is presented as a component for future work in Section 6.1.

5.2 Filter application

In Figures 4.7 and 4.8, it can be noted that the application of all three filters will lead to a reduction of the model's accuracy compared to the baseline. However it should be noted that given the application of the filter the distribution between the positive and negative class is altered in the training data compared to the baseline. The most aggressive reduction of the LRP filter saw a reduction of positive examples in the training data from 19% down to 11%, which corresponds to a reduction of 39.5%. Despite this the accuracy of the model only decreased with roughly 1.25% units compared to the baseline model trained with 39 thousand data samples composed of 19% positive examples.

The application of the filter appears to have a greater effect on the model's accuracy compared to what just reducing the available training data has. However the random removal of training data used for the baseline models did not model the distribution change between the classes which occurred when the three words filters were applied using the sentence dropout method. This can be noted in Figure 4.7. Figure 4.8 shows that the application of the word filters affects the accuracy of the model negatively. However, the negative effect of the application can be considered marginal. Given the possibility of performing informed data regularization by applying the filters, their opportunity cost can be considered very low.

These problems could be avoided by using another filter application solution than the one used in this thesis. However, as mentioned in Section 4.2.1 choosing another application method may lead to other issues and implications than those presented. For example, the issue regarding the risk of changing the meaning of sentences when using a word dropout approach can be considered as more difficult to identify and quantify than the issue of a shifted class distribution in the training data. To be able to quantify this issue of meaning shifting, all sentences affected by the word dropout method would need to be manually inspected for the issue. A task that is dependent on the number of affected sentences could be infeasible due to the scale of the task.

5.3 Responsibility of bias identification and removal

The need for continuous evaluation to ensure that data and models to an extent are bias-free poses the question of how this task should be performed, from both

a method and a responsibility perspective. The methodical perspective of which method is the most effective and how the identification and reduction of biases should be performed, will most probably always be task-specific. Given the broadness of the NLP, this will cause the methodical perspective of this issue to never have a ground truth which is common to the entire field of NLP.

The second question, on who does the responsibility to monitor the bias identification and bias removal lie on? The answer to that question must be the developer or team that built the analysis system. The end-user of a product cannot be assumed to know or be familiar with the specific technology in it, or to exactly know how each component of it works. This could be compared to a customer being required to have a deep understanding of a car's powertrain before being allowed to purchase it. The customer must be made aware of the limitations of a system, but the identification and correction of the limitations must be performed by the system's developers.

Another argument as to why the responsibility must lie on the developers can be viewed in Figure 4.10. The identification of the words *pakistan* and *pakistani* show that there is a clear need for an iterative process of investigation to ensure that a system or data set does not base its prediction on bias terms. However, this process cannot be considered as suitable for complete automation due to the need for analytical decisions based on the core values of the team, developer, or company responsible for the system.

5.4 Identification of a spurious correlation

The general keynote from the results presented in Section 4.3 is that all the three presented methods or bias identification show the ability to produce word filter which can regularize the data such that the impact of specific keywords is reduced. This can be noted in Table 4.8b, 4.9a, and 4.9b. It can be noted from the tables the model weighting for the specific keyword is eliminated and the prediction of the filtered models is flipped compared to the baseline models. However, for example text 3, which has label 1, the application of the three word filters has flipped their predictions compared to the baseline models. This shows that the presented filtering process does not guarantee that the model will correctly classify all sentences containing a keyword just because the relevance of the keyword has been reduced for the model's prediction. Another possible cause for the misclassification could be that the data has been over-regularized such that the model has been able to make a correct generalisation regarding the content of the sentence and its label.

A point of interest that can be noted regarding the filter sourced from the LRP method, presented in Tables 4.6 and 4.7, is that it was possible to reduce the impact of both keywords even though both were missing from the produced word filter. This result which at first appears to be counterintuitive, given that neither keyword is actively filtered from the data, indicates that the keywords must occur in the same sentences as one or several of the other words included in the filter. This is be-

cause the chosen filter application method, sentence dropout, which cause sentences containing the keywords be dropped from the training data without keywords being included in the filter.

Another point of interest in Table 4.6 and 4.7 is that the keyword was identified as among the highest relevant word for true positive classifications. This can be considered as expected given the method of how the training data was augmented in the experiment, as described in Section 3.8. Since all sentences which contained either keyword were moved to the positive class in the training data the correlation between keywords and the positive class was made by the model. This indicates that the model was indeed made to overfit on the specified keywords. This points towards that the method used for data augmentation used in the experiment is sound for its intended goal. However, the lack of identification of the keywords in the LRP filter indicates that the methodology for the selection of the words to filter out can be improved when using the LRP approach.

The results presented in Table 4.10 could point toward that the application of the filters on the training data has broken the induced spurious correlation. The three filtered models do not show the same systemic classification error towards the positive class which was noted in the two baseline models. An interesting take away from the classification results is that the models after application of the word filter the model showed a tendency of favouring the negative class for their prediction. This is noteworthy given that the labels of he sentences containing the keywords are unchanged by the application of the filters. This could indicate that the models after the application of the filters have made a correlation between some content in the sentences containing the keywords and the negative class. If this is the cause for the tendency towards negative classification for the filtered models then the correlation must be stronger than the connection between the sentences containing the keywords and their own label. This could thus indicate that the models have overfitted on a subset of new features, but this time to the negative class instead of the positive.

An important note, regarding the importance of continuous evaluation of the weighting of possible bias inducing words in the model, can be viewed in Figure 4.10. In the figure, it can be noted that all three identification methods manage to remove the keywords from a model's highest weighted words for true positive classifications. However from the model trained with data augmented by the NCOF method the words *pakistan* and *pakistani* have appeared. The appearance of the two words shows that just because the filter methods can help to identify, and through their application reduce the weighting of spurious correlations, that the data or model now are free of any other spurious correlations. This change of focus in the model shows that the elimination of spurious correlations cannot be fully automated as the elimination of one bias can lead to another.

5.5 Comparison of method quality

The three different methods for filter creation all aim to solve the same problem, and even though they managed to produce filter with comparative content there are some qualitative differences between the approaches. The LRP approach is much more computationally heavy than the NCOF and TF-IDF method. This makes the usage of the LRP approach more difficult to motivate for projects that use parameter rich models and expansive data sets, unless infrastructure appropriate for this kind of calculations can be applied. It could possibly be considered that the TF-IDF approach manages to produce a slightly more rich filter in terms of content origin compared to the NCOF method. However, this is a bold claim considering that the result from the two methods is dependent on the evaluated data and this thesis has only investigated one data set. The three methods all produce results that are considered to be of similar general quality and no specific domain is identified to favour any method in particular.

5.6 Discussion regarding the goals and aim of the project

The hypothesis under investigation, as outlined in Section 1.2, and its extension are considered to have been investigated during the course of this project. The methods presented to aid the identification of spurious correlations and the application for removal of spurious correlations have been shown to answer the posed questions.

The aim of the project, to present methods that can help a developer with the task of identifying possible spurious correlations by presenting relevant information are considered to be partially reached. Two of the methods were able to identify the disparity posed by the experiment outlined in Section 3.8, by identifying the specified keywords among the 3σ outliers. However, since there are other types of disparities that can occur it cannot be said that the proposed methods can detect all possible variations.

The presented methods increase the availability of techniques that can work as an aid to a human in the task of detecting spurious correlations. The methods cannot be considered to fully solve the task of identifying a spurious correlation since the act of deciding what is a spurious correlation must be made by a human. The methods merely present present suggestions. However, as discussed in Section 5.3 processes should never be fully automated and the developer should always bear the responsibility of the process.

The goal of the project as described in Section 1.2 are considered to be reached. The presented methods present the information regarding how the data has affected the model's classification and can detect specific words. The methods are also considered to not be content-specific and can present objective results for a broad subject of possible biases.

5.7 Discussion regarding objectives

The questions that this thesis has investigated, previously presented in Section1.3, are presented once again to ease the referral for the reader.

- What information should be presented to a developer such that an informed decision can be made regarding if a spurious correlation have been identified?
- What methods can be used to produce this information?
- How is the presented information weighted in a model during classification?
- Are the methods sufficient to replace current processes for the identification of spurious correlations?

The results presented in Section 4.3 and the question asked in Section 1.3, also presented above, regarding what information should be presented to a developer so a decision regarding the presence of a spurious correlation can be made, are considered to have been answered. The results point towards that two methods were able to propose a spurious correlation in the environment outlined in Section 3.8. The third method, the LRP approach, was however not able to propose the spurious correlation using the specification of word filter creation outlined in Section 3.4.4. The second question in Section 1.3, what methods can be used for the identification, and what are their limitations. Is considered to have been sufficiently answered in Chapters 2 and 3, the theory and method chapters of this thesis. The limitations of the methods, such as their inability to process inflections of words, are problems that possible future extensions of this project may solve.

Regarding the question asked in Section 1.3, if the methods are sufficient to replace current processes for bias identification, the answer is not clear. This thesis did not focus on making any comparisons between the proposed methods and current processes for identifying spurious correlation. The results indicate that the proposed methods when applied as a word filter on the training data using the sentence dropout technique can alter which correlations a model makes between the training data and its labels. However, with a small reduction of the model's accuracy as a result of the alteration and reduction of the training data, it should also be clarified that the methods do not need to be applied as a word filter to be useful, just the identification process can be used to gain insights regarding the content of the data or the weighting of words in a model. This makes the methods a set of tools that can be deployed to work in parallel with any current tools a developer might have available.

5.8 Ethical notes of the project

The project as a whole is considered as ethically sound given that the aims and goals are to increase fairness in machine learning applications and make these processes available to a broader mass.

As mentioned in Section 2.1 the notion of what is considered fair and ethically sound is a subjective matter. This stance in conjunction with the possibility of manual inspection of the results from the method to select the words to be disregarded for a model's prediction can result in biases being overlooked due to a user's personal views. The dilemma of what the most ethical solution should be when a fully automated solution is compared with one involving a human in the loop. Given that the automated solution would have the possibility of making objective decisions based on statistics but would lack the ability to generalise the way a human can, a humans decision process could always be assumed to be affected by the persons own opinions and views.

The experiment which was outlined in Section 3.8 can be considered by some as an unethical experiment especially targeted towards a specific religious group. The experiment did not aim to be disrespectful or to single out any religious groups. The experiment only strives to highlight the issue of skewed data set as the underlying cause for spurious correlations and to investigate if the proposed methods are able to identify and reduce them. Therefore are the experiment considered in the scope of this project as ethically sound.

As mentioned previously in this section, and Section 2.1, what is considered as an ethical bias is subjective. The definition of the term which has been used in this thesis has deliberately been very broad to include as much as possible. This distinction was made to counteract the possible issue of having a too narrow definition which could have led to the issue of underrepresenting the scope of the issue of bias in data.

An ethical note regarding the aim and goal of the project is that if this study is successful in creating a method that can aid a user in reducing biases in a model it can also be used to induce them. This could be done by iteratively evaluating modifications of the model and the training data set until the method detects the sought after bias. The author of this study does not support this use case nor the idea of consciously creating classifiers with a bias.

6
Conclusion

This chapter will give a brief conclusion to the thesis as a whole. In addition to the conclusion, possible future extensions of this thesis are presented.

6.1 Future work

From Figure 4.4 and 4.5 it is possible to see that the padding token used to extend all sentences in the data set receives an LRP score during the evaluation of the model. This indicates that the presence of the tokens is used for the model's classification. From the figures, it is possible to identify that the token always influenced the model to push for a prediction of the negative class. This indicates that the presence of padding tokens in a sentence will push the classifier to classify the sentence as negative. An extension of this thesis could be to investigate this behaviour in the classifier and see if changing the number of padding tokens in a sentence indeed influence the classification.

One possible way to continue this thesis, which has been discussed previously, is to expand the capabilities of the filter production techniques by introducing the ability to process word stems or lemma. The implementation of stem and lemma processing can make the identification more general such that suffixes of words that may be specific to a data set do not influence the identification processes.

Another possible approach to extend this thesis is to investigate the ability to develop a standardised measure of how affected a model is by biases and to identify which kind of bias is affecting it. Due to the lack of standardisation in the field of bias identification in NLP a metric like this is needed so that the evaluation of biases between models is more easily comparable.

6.2 Conclusion

The results presented in Chapter 4 and discussed in Chapter 5 indicate that at least the proposed NCOF and TF-IDF methods can aid with the identification of a spurious correlation that has been injected in a data set, given the environment outlined in Section 3.8. The proposed LRP method did not manage to identify the pre-specified bias but should be able to by making small alterations to the defined method. By applying the three filters to the training data using the sentence dropout method the weighting of the specified keywords was able to be reduced in the model.

The drawbacks of the proposed method limit their applicability. However, neither of the methods is fixed, and the general methodology and theory proposed for the identification and reduction of spurious correlations using the methods shows potential for further development.

The importance of having an iterative process when evaluating the weighting of a model's input has been shown in this study. It was shown that by identifying and reducing the impact of one bias another possible bias term appeared. This also points towards the importance of having a human in the loop to supervise the evaluation.

This thesis by itself cannot be considered as a holistic solution to the issue of bias in NLP data and models. And neither of the proposed methods manages to solve the issue of separating spurious from true correlations or present a way of describing the exact topic that a data set model. However, this thesis proposes three novel methods that aid in making the identification of possible biases and spurious correlations easier. The final decision of regarding what is considered as spurious or not is still a humans, given that it has been shown that this process is not suitable for automation, yet.

Bibliography

[1] Rob Robinsson, "mlnotebook.github.io."

[2] Y. Belinkov and J. Glass, "Analysis Methods in Neural Language Processing: A Survey," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 49–72, 2019.

[3] J. Li, X. Chen, E. Hovy, and D. Jurafsky, "Visualizing and understanding neural models in NLP," in *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference*, pp. 681–691, 6 2016.

[4] H. Liu, S. Shah, and W. Jiang, "On-line outlier detection and data cleaning," *Computers & Chemical Engineering*, vol. 28, 8 2004.

[5] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, no. 7, pp. 1–46, 2015.

[6] A. Datta, M. C. Tschantz, and A. Datta, "Automated Experiments on Ad Privacy Settings," *Proceedings on Privacy Enhancing Technologies*, vol. 2015, no. 1, pp. 92–112, 2015.

[7] S. Kiritchenko and S. Mohammad, "Examining Gender and Race Bias in Two Hundred Sentiment Analysis Systems," pp. 43–53, 2018.

[8] Dastin J, "Amazon scraps secret AI recruiting tool that showed bias against women - Reuters," 2018.

[9] A. Søgaard and A. Johannsen, "Robust learning in random subspaces: equipping NLP for OOV effects," no. December, pp. 1171–1180, 2012.

[10] H. Jiang and O. Nachum, "Identifying and Correcting Label Bias in Machine Learning," 2019.

[11] L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek, "Explaining Predictions of Non-Linear Classifiers in NLP," pp. 1–7, 2016.

[12] Y. Yang, V. Tresp, M. Wunderle, and P. A. Fasching, "Explaining therapy predictions with layer-wise relevance propagation in neural networks," *Proceedings - 2018 IEEE International Conference on Healthcare Informatics, ICHI 2018*, pp. 152–162, 2018.

[13] D. Shah, H. A. Schwartz, and D. Hovy, "Predictive Biases in Natural Language Processing Models: A Conceptual Framework and Overview," no. 2003, 2019.

[14] K. Kurita, N. Vyas, A. Pareek, A. W. Black, and Y. Tsvetkov, "Measuring Bias in Contextualized Word Representations," pp. 166–172, 2019.

[15] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K. R. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE Transac-*

tions on Neural Networks and Learning Systems, vol. 28, no. 11, pp. 2660–2673, 2017.

[16] J. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pp. 6382–6388, 2020.

[17] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," tech. rep., 2014.

[18] K. Lum and W. Isaac, "To predict and serve?," *Significance*, vol. 13, no. 5, pp. 14–19, 2016.

[19] L. Sweeney, "Discrimination in online Ad delivery," *Communications of the ACM*, vol. 56, no. 5, pp. 44–54, 2013.

[20] G. Ford, "4 human-caused biases we need to fix for machine learning," 2018.

[21] E. Notes, "Artifical Neural Networks B. M," vol. 26, 2019.

[22] T. Wood, "Convolutional Neural Network Definition | DeepAI."

[23] F. Chollet and others, "Keras." \url{https://keras.io}, 2015.

[24] T. M. Cover, "Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition," *IEEE Transactions on Electronic Computers*, vol. EC-14, no. 3, pp. 326–334, 1965.

[25] F. Harrag, "Event Extraction Based on Deep Learning in Food Hazard Arabic Texts,"

[26] R. Pryzant, R. D. Martinez, N. Dass, S. Kurohashi, D. Jurafsky, and D. Yang, "Automatically neutralizing subjective bias in text," *arXiv*, no. 2013, 2019.

[27] J. Vaughan, A. Sudjianto, E. Brahimi, J. Chen, and V. N. Nair, "Explainable Neural Networks based on Additive Index Models," no. June, pp. 1–11, 2018.

[28] A. Binder, S. Bach, G. Montavon, K. R. Müller, and W. Samek, "Layer-wise relevance propagation for deep neural network architectures," *Lecture Notes in Electrical Engineering*, vol. 376, pp. 913–922, 2016.

[29] A. Binder, G. Montavon, S. Lapuschkin, K. R. Müller, and W. Samek, "Layer-wise relevance propagation for neural networks with local renormalization layers," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9887 LNCS, pp. 63–71, 2016.

[30] P. Rayson and R. Garside, "Comparing corpora using frequency profiling," pp. 1–6, 2000.

[31] Juan Ramos, "Using tf-idf to determine word relevance in document queries," *Proceedings of the first instructional conference on machine learning. Vol. 242. 2003.*, vol. 2, no. 1, pp. 29–48, 1975.

[32] G. Salton, "1988_Salton, G. and Buckley, C., 1988. Term-weighting approaches in automatic text retrieval._7896.pdf," 1988.

[33] A. Berger, R. Caruana, D. Cohn, D. Freitag, and V. Mittal, "Bridging the lexical chasm: Statistical approaches to answer-finding," *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, pp. 192–199, 2000.

[34] E. L. Bird, Steven and E. Klein, *Natural Language Processing with Python*. O'Reilly Media Inc, 2009.