Anti-Spam : A new methodology for interpreting anti-spam classification
Project 2 Report, NLP Course, Winter 2022

Marcin Łukaszyk
WUT
Student
01133055@pw.edu.pl

Jean-Baptiste Soubaras
WUT
Student
01182889@pw.edu.pl

Supervisor: Anna Wróblewska
WUT
anna.wroblewska1@pw.edu.pl

Abstract

The war on spam e-mails has been an important issue in the recent decades, and the development of Machine Learning (ML) methods - especially that of Recurrent Neural Networks (RNN) - in the Natural Language Processing (NLP) domain led to always more accurate AI-based filters able to classify e-mails are *spam* or *ham*. In return, the behavior of these algorithms has been more and more opaque for human understanding, giving very few insights on the resolution of the problem as well as on the limitations of the algorithms. The subject of the project described in this proposal is to research Explainable Artificial Intelligence (XAI) methods to solve the spam detection problem while giving humanly-interpretable insights on the deeper functioning of the ML algorithm. The global purpose would be to elaborate a methodology to make any AI-based classifier more interpretable. To proceed, the project will be divided in two parts : the first aiming at the writing of an efficient AI-based spam filtering algorithm, the second at the exploitation of this algorithm using diverse XAI methods in order to make it more explanatory.

1 Introduction

1.1 Background

The outburst of the Internet in the late 90s was a phenomenon that revolutionised the way of communicating. In comparison with the traditional post mail, the arrival of the electronic mail, more commonly named e-mail, and its wide-spreading use allowed for much quicker and more convenient message delivery. However, such a convenient media would quickly show a few drawbacks. One of them is the overabundance of spam, also called junk mails.

A spam is an irrelevant or unsolicited message sent by mail, generally to a large number of users, for the purposes of advertising, phishing (i.e. tricking someone into giving access to their credentials or bank data for malicious purpose), spreading malware, etc. It is estimated that the proportion of spam among the global em-mail traffic is about 50%. It has thus been an issue for mailbox providers to ensure that their users would not be exposed to such content, by filtering the incoming messages and programmatically spotting the ones suspected to be spam, generally to redirect them to another box labeled as ”SPAM”. Most of the algorithms assigned to this task use Machine Learning (ML) methods to achieve their goal.

The main problem caused by ML algorithms is that they act like a black box, able to classify a given input but without being able to give humanly-interpretable reasons for their choice. As a result, the algorithm is unable to provide an understanding on the problem, in spite of its ingestion of large sets of data. In the recent years, many researchers have worked on the interpretability of such algorithms, in order to find methods for using the ability to train on a large number of data as a way of obtaining a better understanding on a problem.

The problem of spam detection is quite adapted for tackling this issue for two reasons : first, it is a problem that modern ML algorithms tackle quite easily (for instance, *Google* and its mailbox *Gmail* claim an accuracy of 99.9% of their spam and phishing detection algorithms); second, human analysis and understanding already give a lot of insights on the characteristics that make an e-mail

a spam e-mail. Actually, a lot of spam senders display some hints in their message (spelling mistakes, unrealistic promises,...) in order to target the less vigilant people that will follow their instructions until the end and are less likely to engage in judiciary process). It is thus a good topic for implementing interpretation methods for ML algorithms and analysing their outputs.

1.2 Description of the subject

The main goal of the project is to analyse and explain predictions of machine learning models resolving the spam detection problem. Knowing how different inputs affect outputs and why output changes helps build trust and brings AI closer to general audience. Moreover XAI (Explainable artificial intelligence) can be used by researchers to confirm existing domain knowledge or to discover new insights. It can also be used to look for any existing bias in machine learning models.

1.3 Significance of the project

Applying deep learning techniques to NLP problems had greatly improved scores and ability to solve complex task. Contrarily it had negatively affected how attainable is to interpreted inner workings of complex machine learning models. To counter lost information it is necessary to ellaborate methods of making models more explainable and interpretable. During our work we will try to use techniques like: feature importance, surrogate model and visualizations of model predictions.

Spam and spam detection started as soon as internet become something publicly available in mid 1990 although first spam email is dating back to May 3, 1978. First methods of dealing with spam were basically blocking given addresses. Mainly in form of black hole lists with blacklisted addresses. Other methods were based on DNS addresses. During 2000 spam became serious problem as it impacted user experience and many computer viruses were transformed using spam emails. To fight it many organizations (public and private) created various algorithms and legal measures. They used techniques like ham passwords, Checksum-based filtering or first NLP methods based on regular expressions.

The global improvements in spam filtering algorithm have led to a decline in the total amount of spam sent each year.

Interpretability and explanability is vital for further as making sure how and why given algorithms work help ensure credibility of reserchers. To keep up with new, more advanced methods and algorithms for NLP researchers must create and improve various methods to explain models behavior.

2 Related Work

2.0.1 Current algorithms

Most of machine learning algorithms can be applied to task of spam detection. Most popular ”standard” ones are KNN, Naïve Bayes and Reverse DBSCAN (Harisinghaney et al., 2014) or random forest. More advanced models using deep learing are also in use like standard deep learning neural networks, Long Short Term Memory networks (AbdulNabi and Yaseen, 2021) or Convolutional Neural Network and Multi-Layer Perceptron (Shahariar et al., 2019).

2.1 Future perspectives for neural networks interpretability

Interpretability and explanability is vital for further as making sure how and why given algorithms work help ensure credibility of reserchers. To keep up with new, more advanced methods and algorithms for NLP researchers must create and improve various methods to explain models behavior.

2.1.1 Methods employed

(Belinkov and Glass, 2019) provides a benchmark on several methods used for interpreting classifiers based on neural networks. Many methods rely on the prediction of linguistic properties from activation of the neural network. In this approach a first neural network model is trained on the main problem. Then, the trained model is used for generating feature representations. Finally another classifier is used to predict the linguistic property of interest.

Another approach suggests linguistic Correlation Analysis and Cross-model Correlation Analysis with comprehensive analysis of neurons to analyse distribution of different linguistic properties and neurons exclusivity to some properties.

First method, linguistic Correlation Analysis, trains second, some kind of linear model for easy explanability, based on neuron activations values from first, already trained on our dataset model, with labels from our original dataset. Then based

on absolute values of weights in new, second model we can deduce ranking of neurons importance of first model, trained to detect spam.

Cross-model Correlation Analysis works by training multiple similar ("using identical model settings but with differing training data and initialization") models and then for each neuron in our architecture compering Pearson correlation coefficient between neuron activation values in original models and ones without neurons. Then we can rank them based on correlation coefficient and deduce most important ones.

2.2 Data Sets

A preliminary research allowed to find several available data sets.

SpamAssasin - 2001 / 6047 instances
This data set has been created by the *Apache Software Fondation* to develop their spam filter software.

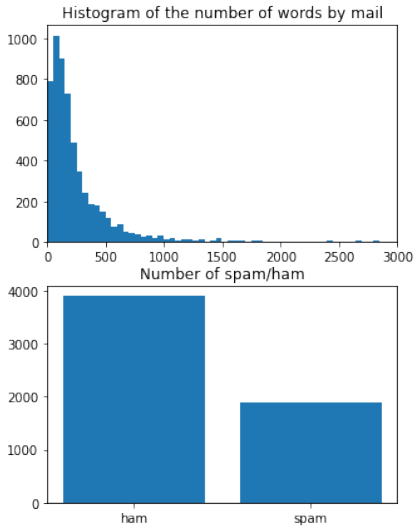
Enron-Spam - 2006 / 6000 instances
The data set was created and studied in the following article (V. Metsis and Paliouras, 2006).

SMS Spam Collection - 2012 / 5574 instances
The data set was created and studied in the following article (T.A. et al., 2011). It contains SMS and not e-mails, but it will be used to compare the interpretations obtained on different formats of text.

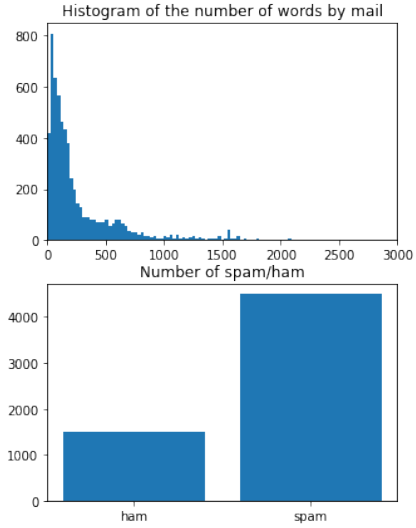
Two additional data sets dedicated to this end were also found, however for privacy reasons these data sets were already encoded and are not suitable for language interpretation. In consequence, they will only be considered for testing purposes if needed.

Spambase dataset - 1999 / 4601 instances
The data set was created by (Hopkins et al., 1999). The file contains a classification based on 57 features of e-mails, most of them being the frequency of a word or character. There is no acces to the raw text data.

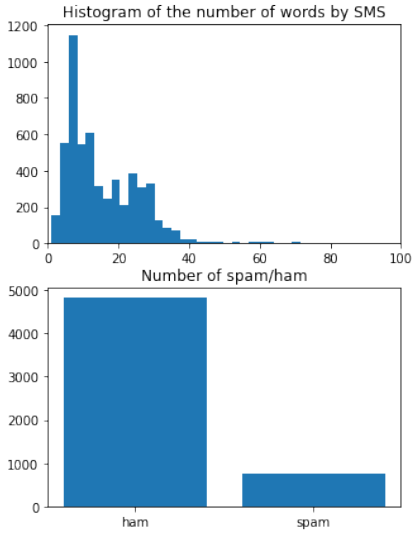
PU Corpora - 2003
The data set was created by (I. Androutsopoulos, 2003). The e-mails contained in the data set are all tokenized for privacy matters.



(a) Spamassasin



(b) Enron



(c) SMS Spam Collection

Figure 1: Content of the data sets.

3 Approach and Methodology

3.1 Functional methodology

1. Preliminary bibliographical research
2. Creation of a work environment
3. Implementation of simple spam detection models
4. Evaluation and analysis of the spam detection
5. Implementation of known methods for interpretation of the model
6. Evaluation and analysis
7. Research and improvement -¿ added value of our work
8. Evaluation and analysis
9. Report and presentation

3.2 Evaluation and analysis

The different methods that will be studied have been detailed in section 3. Considering the primary spam detection data set, that latter will be evaluated on a test data set. The purpose is to use neural networks to train an efficient enough model, yet opaque in its execution.

For XAI algorithms, evaluation will be made by comparison of the interpretation computed with human interpretation and comparison of the output with that of the primary algorithm. The project will use an additional data set consisting of SMS spam to compare the differences of interpretation between e-mail spam and SMS spam. Eventually, if relevant enough, an implementation of naive heuristic methods based on the insights given by the XAI methods could be compared with the primary algorithm.

3.3 Tools

The tools that will be used will be detailed in this part.

Hardware

- Personal computers;
- In case of a need for heavy computation (training ML models on large data sets), the use of the computers of the laboratories of WUT could be envisaged.

Software

- Programation language: Python 3.6;
- Libraries: Natural Langueage Toolkit, Keras, Tensorflow;
- Collaboration: Google Collab, maybe GitHub;
- Data sets: Kaggle connector;

4 Preliminary work and classification

4.1 Exploratory Data Analysis

For the Exploratory Data Analysis (EDA), we focused on the two main data sets : *Enron* and *SpamAssassin*. The data sets were first parsed and preprocessed to remove all punctuation, digit and stop word, and to lemmatize all the remaining words. Then, in Figure 2, the word cloud of each data set was plotted according to the class of each mail : ”spam” or ”ham”.

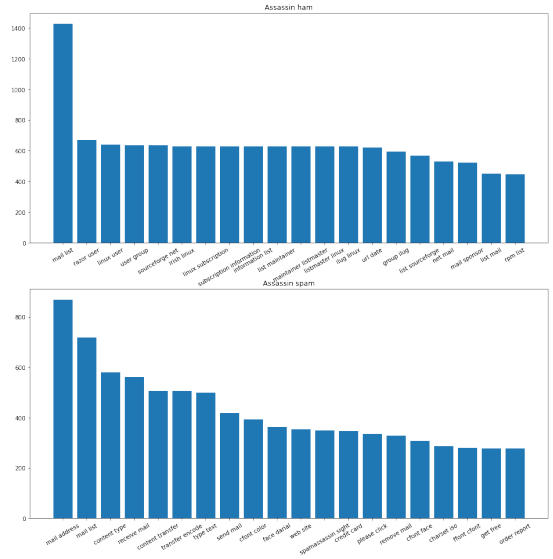
(a) Spamassasin

(b) Enron

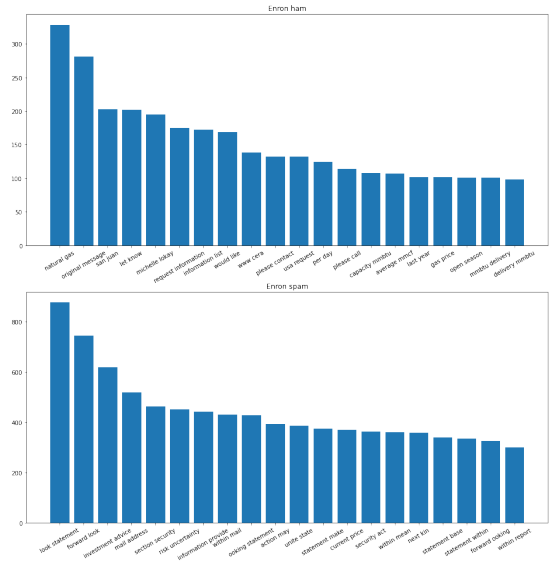
Figure 2: Word clouds of the data sets.

We can observe first that the Enron data sets uses much more words coming from the professional language (”market”, ”company”, ”employee”,...) whereas in SpamAssassin the most common words are words of the everyday language. Moreover, in both cases the spam mails seem to use less of these overly employed words, but instead contains a lot of occurrences of expressions that would in a lot of cases seem suspicious (for instance, very few ”investment advices” are given by mail).

To get more first hand insights, the repartition of the most common bigrams were also plotted in Figure 3.



(a) Spamassasin



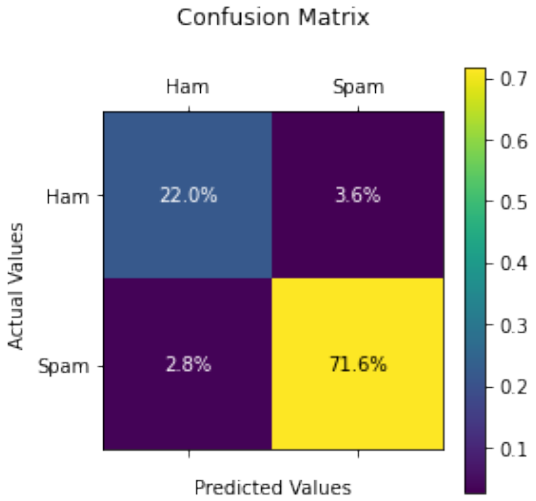
(b) Enron

Figure 3: Most common bigrams.

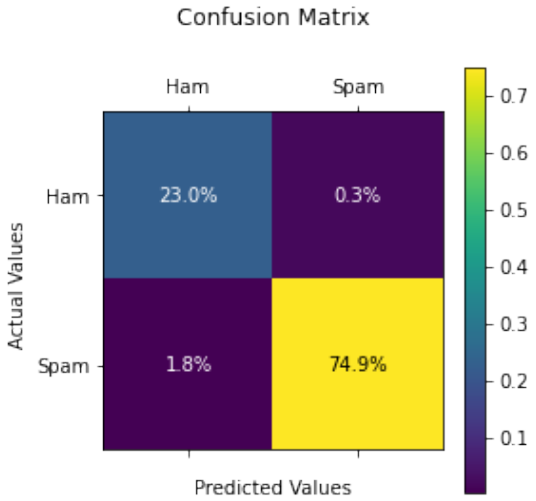
Again, the vocabulary is much more professional in the Enron data set, and the choice of words differ quite a lot between spam and ham mails.

4.2 Classification

The task of classification not being the heart of our research work, we implemented a generic neural network model (the sequential model from the library `keras`). For the sake of comparison and in to attest the quality of the model, we also implemented a Naive Bayesian algorithm on CountVectorizer embedding, which is a classical algorithm for solving our problem explained in (Manning et al., 2008). Figure 4 relates the performances of both classifier. Our Neural Network (NN) is slightly more efficient than the Naive Bayesian classifier, with a score of 98% accuracy.



(a) Count Vectorizer + Naive Bayesian



(b) Neural Network

Figure 4: Confusion matrix of each classifier.

The model we trained is thus suitable to be analysed using different XAI methods.

5 XAI methods and results

5.1 Linguistic Correlation Analysis

The linguistic correlation analysis method is used to extract most important neurons in a given task. To do that we need to find a method of ranking all neurons and then to visualize it.

Ranking of all neurons is done by training logistic regression classifier on a new ”meta” dataset. We create this dataset from testing set of our original dataset. We fit each record to our already trained model (on training dataset) and record each neuron activation value. So for each training record in original dataset we get vector of values representing every neuron in every layer. We delete values from last neuron as it is a binary classification and last layer is a softmax layer and would provide too much information to the logistic regression model. We take the same labels as in original set and just write them to our new dataset.

Then we train a logistic model on new ”meta” dataset in the same task of predicting spam messages. This time we use neuron activation values and that's why we did not include values from the last layers as they have most direct effect on the prediction that highly outweighs other neurons. Problem with the disproportional ”power” of neurons in last layers is acknowledged further down.

We use logistic regression as it is a highly explainable model. In our example, we fit our ”meta” dataset where each value in input data corresponds to each individual neuron in neural network and has a direct impact on value of the corresponding β parameter from coefficients of logistic regression model. This way we created a relationship between neuron in neural network and the coefficient of logistic regression where the absolute value of coefficient directly affects its voting power in prediction.

This way we achieved ranking of neurons based on absolute value of corresponding coefficient in neural network.

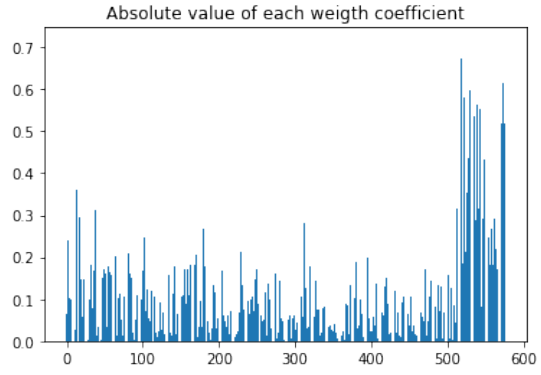


Figure 5: Distribution of absolute value of coefficients for each neuron

Testing if this method of ranking works is done by observing how performance of original model changes when we will be only using some chosen subset of neurons based on our ranking. In our experiment we reevaluate performance of orginal model when using:

- Top 20% of nurons
- Next 20% (20-40% of ranked neurons)
- Bottom 80% (20-100% of ranked neurons)
- Top 80% (0-80% of ranked neurons)

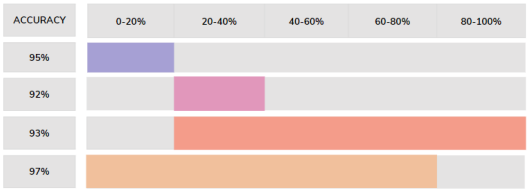


Figure 6: Model accuracy when using selected number of neurons

Table 6 proves that using logistic regression classifier to rank neurons works and sets a general ranking of neuron importance.

Additionally, we tested different way of ranking and picking neurons. As it was stated neurons in smaller and later layers has greater effect on prediction we implemented a method to nulify it. Instead of ranking them all together we rank them individually in each layer. This way we are sure that each layer is represented in equal part and relative size and position dose not affect ranking.

To see how this method affects model we compared accuracy when using:

- 0-20% of ranked neurons
- 20-40% of ranked neurons
- 40-60% of ranked neurons
- 60-80% of ranked neurons
- 80-100% of ranked neurons

When using individual layer ranking we got better accuracy on top-performing neurons. Accuracy next neurons is worse but the overall accuracy of both models drops significantly making both models useless. of ranked neurons

Chosen percentage	One ranking accuracy	Individual layer accuracy
0-20%	0.94	0.96
20-40%	0.79	0.78
40-60%	0.69	0.79
60-80%	0.70	0.59
80-100%	0.70	0.43

Basic visualizations are made by calculating how lack of each word in a sentence affects neuron values. To do that from one input which is sequence of words $\{w_1, w_2, w_3 ... w_n\}$ we create set of n (where n is number of words in sequence) sequences where for each word there is a corresponding sequence **without** it in it. So we got set of $A = [\{w_2, w_3 ... w_n\}, \{w_1, w_3 ... w_n\}, \{w_1, w_2 ... w_n\} \cdots \{w_1$

Then we predict each sequence in set using neural network model and observe how pre-chosen neurons value changes based on missing words. After getting values for each neuron and each word we can visualize it based on change of neuron values.

Figure 7: Example of visualization for top 100 neurons on test set example

Advanced visualizations are made by picking only one neuron and seeing how it reacts to many different sentences or by analyzing one sentence and showing changes of values for each neuron individuality. Then we can show them next to each other and compere results.

Figure 8: One neuron shown against many different sentence

Finally, we implemented a search for a chosen word (e.g. call) in our dataset and implemented visualization of the searched word in a different fashion. Instead of colouring background of the word in green/red depending on its effect on the word we colour words (each letter in the word in the same colour). This way we have an easier way

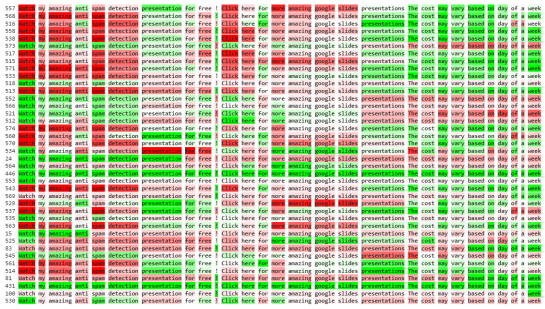


Figure 9: Many neurons show against one sentence

of spotting searched word and can show all sentences in dataset having this word in it.

Figure 10: Many neurons show against one sentence

5.2 Frozen Weights Method

5.2.1 Different Approaches

The frozen weights method consists in freezing the inputs and outputs weights of each neuron of a trained model (see Figure 11) in order to predict the linguistic properties they describe (occurrence of a word, part of speech, meaning-based properties...). Note that the activation biases are not considered here.

To predict linguistic properties from the weights of a neural network, we need a benchmark of neural networks of the same architecture dedicated each to the detection of a specific linguistic property. In this work, we used a family of neural networks trained on the detection of one specific word each, these words being part of the vocabulary of the train data set. Another family of properties we could have used would have been a classification of the mails according to their meaning or sentiment, but it would have been harder to train the models.

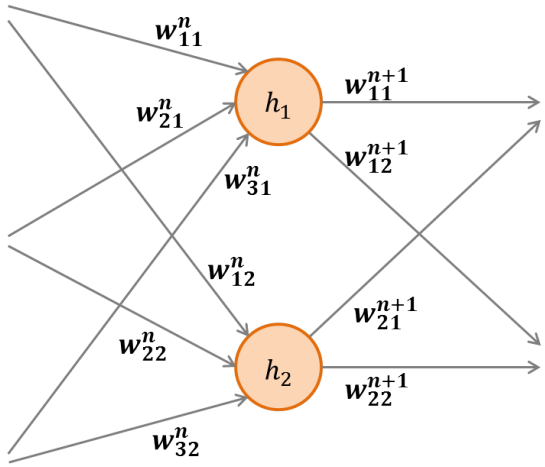


Figure 11: Weights extracted from the neurons.

To compare the weights of the main model with those of the benchmark models, we considered three possibilities:

- Classification;
- Optimization;
- Model embedding.

Classification is the most common approach in the literature (see (**?**)). It consists in training an external classifier (usually deep learning-based) on the benchmark model weights to predict the most accurate linguistic property that relates to the main model. This approaches has shown interesting results in the literature, however it is quite intricate and very heavy in terms of computational costs.

Optimization consists in defining a distance (by default the Euclidean distance) between the weights of two models (with respect to the used architecture). Then, the problem relates to a combinatorial optimization problem aiming at finding one or several models of the benchmark being the closest to the studied model. This allows to find the properties whose detection is a problem similar to the main spam detection problem. The drawback of this method is that combinatorial optimization is often quite intricate and usually needs a heavy use of heuristics.

Model embedding is an approach inspired by the vectorization of word embedding. It consists in flattening all the weights of the given model into a vector of an Euclidean space. Under that form, the corresponding vector can be expressed as a linear combination of the vectors of the embedded models of the benchmark. We can then project the vector into the subspace engendered by a few of the most relevant models. More formally, if M is the studied model, $B_{1:k}$ the benchmark models, $v(\cdot)$ the vectorization operation, and m the number of properties we want to observe:

1. Compute $\{|\langle v(M), v(B_i)\rangle|, \forall 0 \leq i \leq k\}$;
2. Find $i_1, ..., i_m$ the m biggest element of that set;
3. $v(M) \approx \sum_{l=1}^{m} \langle v(M), v(B_{i_l})\rangle v(B_{i_l})$.

If we consider the coefficients associated with each benchmark model as a correlation, we can express the resolution of a problem by a neural network by a combination of the detection of several characteristic linguistic properties.

5.2.2 Optimal Design for Model Embedding

In this project, the approach adopted was the latter. For the approximation $v(M) \approx \sum_{l=1}^{m} \langle v(M), v(B_{i_l})\rangle v(B_{i_l})$ to be considered reasonable, the basis vectors should have three characteristics:

- Be in large number and diverse (*range*)
- Be almost orthogonal (*inter-correlation*)
- Have the biggest coefficient with respect to the main model (*amplitude*)

The purpose is thus to maximize range and amplitude while minimizing inter-correlation. To achieve it, the following methodology was established:

- Choice of the family(ies) of linguistic properties to project the main model on (*preselection*)
- Training and vectorization of the main model
- Selection of the exact linguistic properties to train (*heuristic selection*)
- Training and vectorization of the models for each property
- Selection of the most adapted properties (*optimal selection*)
- Exploitation of the results for interpretation purpose.

Pre-selection : The linguistic properties that can be used need to be as simple as possible and can be only labeled are present in the text or absent. Here are a few examples:

- Presence of a specific word or expression
- Sentiment analysis
- Topic-based classification (commercial, professional,...)
- Part of Speech
- ...

In this work the family considered was the presence of specific words encountered in the vocabulary of the training data set. However, there are tens of thousands of word in the English language, and training thousands of models is time consuming. Thus the need for a first heuristic selection in order to limit the number of models to train.

Heuristic Selection : The heuristic selection need to reduce the number of potential properties to use while keeping a large enough range. It is thus necessary to make use some heuristics to estimate which properties are more likely to be relevant to the result. In the case of presence of a word detection, here are examples of heuristics that can be used:

- Results from the Linguistic Correlation Analysis
- Number of occurrences of the word (or the target property) in the train data set
- Similarity measures using word2vec embedding
- ...

In this project, the first heuristic has been used: the top 20% of the words according to their linguistic correlation analysis score were used to train basis models.

Optimal selection : Once the potential basis vectors are computed, the subset of the final basis vectors has to be determined accordingly to the purpose of minimizing inter-correlation and minimizing amplitude. Here is the method to use for this selection:

1. Let S be the subset of selected vectors. Initially $S = \emptyset$.
2. At each iteration, find the vector from the flattened models that minimizes:

$$\mathcal{L}(w) = \max_{b \in S} |\langle w, b \rangle| - \lambda |\langle w, v \rangle|$$

   where $\lambda > 0$ is a parameter.
3. Add the corresponding vector to S.
4. Repeat until S reaches a target size.

The first term of the loss function represent the inter-correlation, the second the amplitude.

5.2.3 Results and Discussions

This was the methodology the project applied, and Figure 12 displays the results it gave on the model.

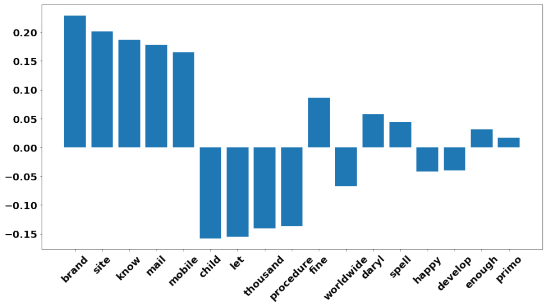


Figure 12: Results of the embedded frozen weights method.

Our model is meant to detect spam mails, so the values of each word refer to the correlation between the problem of detection of a spam mail and that of detecting the specific word in question. Thus positive values indicate that the corresponding words contributes to making a spam mail while negative values contribute to making a ham mail.

It can be observed that there are more positive values than negative, which indicates that the prediction of a mail as ham relies most on an absence of certain words than the presence of some others. This insight is actually quite intuitive, since ham mail have much more variety of purposes than spam mails but the results of our EDA (Figure 2) seemed to indicate that there it would be the opposite, since the vocabulary used in the data sets is more diversified in spam mails.

Another aspect that can be noticed is the distinction on this plot bar of three distinct groups of words: the strongly positively correlated words, the strongly negatively correlated words and the moderately correlated words. This clear distinction indicates that the choice of vocabulary is predominant in differentiating a spam mail from a ham mail, as the words of the first two groups are the ones overused by one of the class and almost not by the other.

When looking at the words that have been selected, it appears that the criteria of low inter-correlation and high range are respected since all these words target different meaning and translate different purpose of the mails. For example, `brand` indicate some commercial purpose in a mail, which indeed is recurrent in spam mails. The word `site` is used to push the receiver to click on a hyperlink, which is also a common tactic used by spam mails. On the other hand, `procedure` shows that the mail has a professional purpose, which justifies his negative correlation. Some of these results are harder to interpret, especially when it comes to very common words, such as `know` which is unexpectedly strongly positively correlated with spam. This insight indicates that spam mails rely on pretending a certain knowledge that the receiver doesn't have. Among the negatively correlated words, `let` is also very common and relates to situation of regulations and authorizations, which are quite common in professional exchanges. Finally, the presence of the word `happy` seems to give some insights on sentimental analysis, even if its negative correlation is probably due more to formal expressions used in professional exchanges rather than the conveying of genuine joy.

5.2.4 Further Perspectives

Since this method uses linear transformations, it can be used to compute the coefficient of a text or subtext, allowing for example to identify sentences or formulations that drive the model towards predicting one class or the other. However, using this approach on larger texts with more diverse vocabulary would require to have more words/properties in the selected results at the end, and increasing the vocabulary means increasing the number of models to train which were already very needy in terms of computational power. Moreover, in order to deal with synonymous expression word2vec embedding could be used to compute their coefficients by projecting the linear relations between words as linear projections into linguistic properties, thus enlarging the vocabulary for computing text correlation.

Finally, what is lacking for a proper evaluation is a metric to quantify the accuracy of the results. This is a hard task since interpretation relates more on quality than quantity, and no method has been found in the litterature other than comparing the results with a manually annotated data set. Unfortunately, no such data set on the spam detection problem has been found, thus improving this method would require either to create this data set or test it on another problem where annotated data sets already exist, both solutions that exceed the scope of this project.

6 Conclusion

This project aimed at first using a neural network model to solve the spam detection problem, then analysing and interpreting that model using state-of-the-art XAI methods. After preliminary data pre-processing and Exploratory Data Analysis, a rather simple neural network based classifier was implemented, evaluated and compared with the Naive Bayesian algorithm to attest its robustness. In the second part of the project, two different XAI approaches were tested : linguistic correlation analysis and frozen weights method.

Linguistic correlation analysis allowed to visualize the impact of each neuron on the output of the model, whereas frozen weights aimed at describing the model as a vector in a space representing linguistic properties. Both allowed to display some relevant results and insights, but which were not quantifiable thus couldn't be evaluated by any metric. If further researches were to be made, they should revolve around two axis:

- Observing the output obtained with more computational power;
- Evaluating the results, by either creating an annotated data set or implementing these XAI methods on an already existing one.

References

[AbdulNabi and Yaseen2021] Isra'a AbdulNabi and Qussai Yaseen. 2021. Spam email detection using

deep learning techniques. *Procedia Computer Science*, 184:853–858. The 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops.

[Belinkov and Glass2019] Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics (TACL)*, 7:49–72.

[Harisinghaney et al.2014] Anirudh Harisinghaney, Aman Dixit, Saurabh Gupta, and Anuja Arora. 2014. Text and image based spam email classification using knn, naïve bayes and reverse dbscan algorithm. pages 153–155, 02.

[Hopkins et al.1999] Mark Hopkins, Erik Reeber, George Forman, and Jaap Suermondt. 1999. Spambase data set.

[I. Androutsopoulos2003] E. Michelakis I. Androutsopoulos, G. Paliouras. 2003. Learning to filter unsolicited commercial e-mail. .

[Manning et al.2008] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. .

[Shahariar et al.2019] G. M. Shahariar, Swapnil Biswas, Faiza Omar, Faisal Muhammad Shah, and Samiha Binte Hassan. 2019. Spam review detection using deep learning. In *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 0027–0033.

[T.A. et al.2011] Almeida T.A., Gomez Hidalgo J.M., and Yamakami A. 2011. Contributions to the study of sms spam filtering: New collection and results. *2011 ACM Symposium on Document Engineering (DOCENG'11)*, pages 1–9.

[V. Metsis and Paliouras2006] I. Androutsopoulos V. Metsis and G. Paliouras. 2006. Spam filtering with naive bayes - which naive bayes? *3rd Conference on Email and Anti-Spam (CEAS 2006)*.