

Università degli Studi di Milano Bicocca



Text Mining Project Report

By

Castagna Alessandro

Colombaro Daniel

Pasotti Matteo

SMS Spam Text Classification and Summarization

MSc Data Science

Tuesday 6th February, 2024

Contents

1	Introduction	1
1.1	Introduction	1
2	Classification	4
2.1	Data Preparation	4
2.2	Bidirectional LSTM	5
2.2.1	Structure of the network	7
2.3	BERT Transfer Learning	8
3	Topic modeling	10
3.1	Latent Dirichlet Allocation	11
3.2	Abbreviation Expansion	12
3.3	Evaluation	14
3.3.1	Interactive Visualization of Topic Distribution and Saliency	15
4	Conclusions	20
4.1	Classification	20
4.2	Topic Modeling	21

Chapter 1

Introduction

1.1 Introduction

In this project, we implemented various neural networks to identify *spam* among a dataset consisting of SMS. The goal was to correctly identify *spam* messages. Furthermore, we applied **text summarization**¹ to derive the key common features among spam emails and how they differ from the main characteristic of '*normal*' messages.

The dataset consists of 5572 observations with two features:

- **Text:** The textual content of the SMS.
- **Target:** The ground-truth classification: it attains value '1' if the observation is labelled as *spam* and '0' otherwise.

The **target** feature presents a skewed distribution, as the majority of the observations are considered normal, as Figure 1.1 shows.

¹The process of producing a restricted version of a text that contains information that is important or relevant to a user.

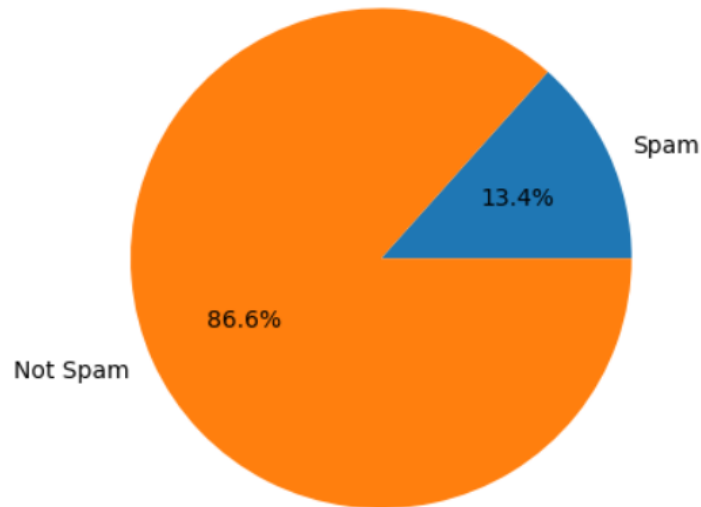


Figure 1.1: Pie Chart

The sentence length of the '*Text*' feature can be informative to determine the label of the observation. In fact, normally messages are used for quick conversational exchanges with a limited number of words, while spams are generally longer. This expectation is supported by evidence in Figure 1.2.

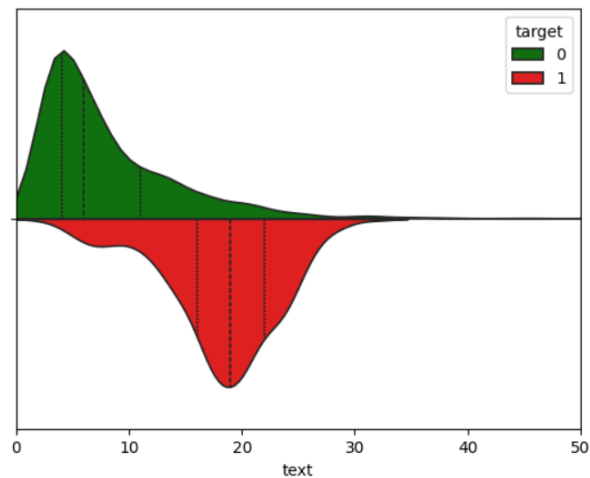


Figure 1.2: Violin Plot

The distribution of normal messages revolves around 5 words per message with some right outliers, while spams have a peak at almost 20 words, confirming the previous assumption.

A visual overview of the most common words that appear in spam messages is provided by the word cloud in Figure 1.3

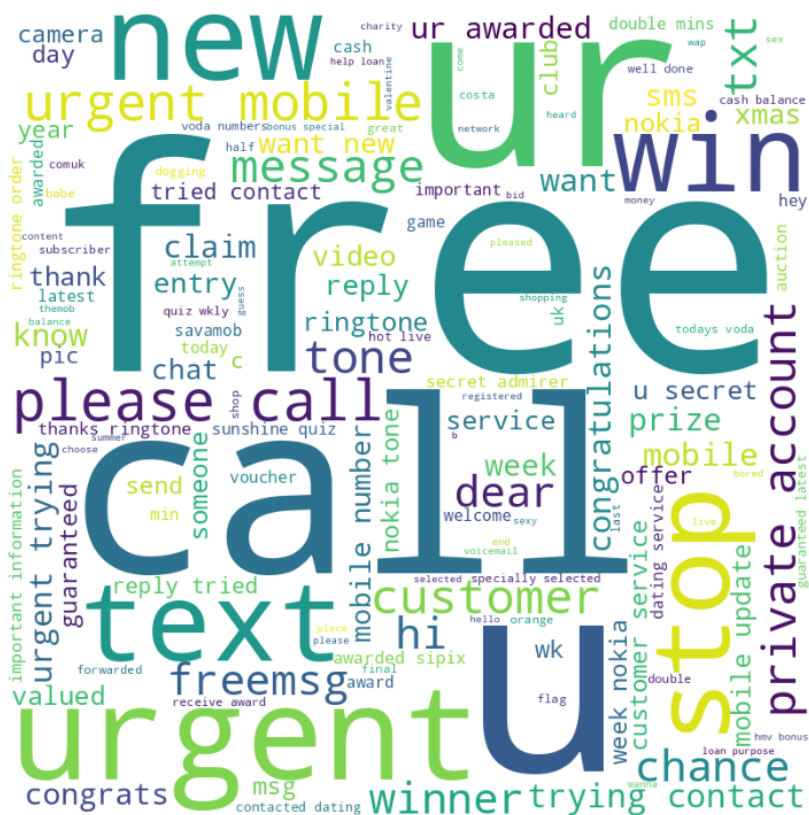


Figure 1.3: Word Cloud of spam messages

The most frequent terms instill a sense of urgency and opportunity to win something for free, with various *call-to-action* words.

Chapter 2

Classification

2.1 Data Preparation

We have performed some adjustment operations on the textual data to remove noise and filter the informative part. Firstly, we removed special characters and any sort of punctuation as these do not provide any semantic meaning, but only serve a grammatical purpose. As a result, only alphanumerical characters were kept.

Luhn's heuristic method is one of the early approaches of text summarization. It argues that the importance of a document depends on the present of **significant words**, where the latter are words whose occurrence is median, while **stopwords** have the most occurrence and **tailwords** the least occurrence. Following this idea, we have imported the package of all English stopwords (the language of the SMS is always English) and removed this list of words from the messages, as they are too common to provide any meaningful pattern.

The following step in the transformation process of text sequences is **tokenization**, which was done in accordance with the classifier associated to it.

2.2 Bidirectional LSTM

Our first classification approach consisted of developing a **Bidirectional LSTM**[1] recurrent neural network. Before creating the model, the text was tokenized using [Keras Tokenizer](#), which vectorizes a text corpus, by turning each text into a sequence of integers representing the indices of tokens in a reference dictionary. The index '0' is not assigned to any word, as it is reserved for padding, while the lower indices represent more common words. To avoid information leakage, data have been previously split into training and test set, and the *Tokenizer()* was fit on the training part only and applied to the whole data.

The output of the *Tokenizer()* is a list of integer numbers for each observation, but the required input for neural networks is a multidimensional tensor with fixed dimensions. The size of each list of indices depends on the number of words in the original sentence and consequently varies. We used *padding* to fix the length and we set the threshold to the 75th-percentile of the length distribution among the training observations, which corresponds to 15 (15 words per message). As a result, longer messages have been truncated and shorter messages have been padded with '0' to reach the desired sequence length.

Recurrent Neural Networks (**RNN**)[2] differ from feedforward neural networks by the presence of feedback connections where the flow of information occurs between neurons of the same layer or from higher layer neurons to lower layer neurons. In the context of text modeling this means that the value of a certain neuron $x(t,n)$, representing the " t -th neuron at layer " n ", is derived not only from neuron $x(t,n-1)$, which is the same neuron in the previous layer, but also from neuron " $x(t-1,n)$ " which is the previous neuron of the same layer, which represents a derivation of the embedding vector of the anterior word in the sentence.

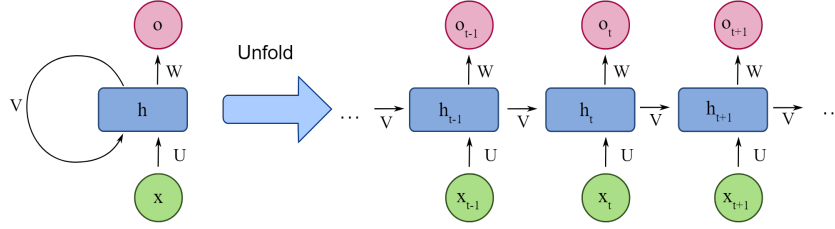


Figure 2.1: Recurrent Neural Network

The memory unit, also referred to as the LSTM[3] unit, is the only component of the LSTM architecture. There are four feedforward neural networks that comprise the LSTM unit. The three common memory management functions are carried out by these three gates: the forget gate, which removes information from memory, the input gate, which adds new information to memory, and the output gate, which uses the information already there. To generate fresh candidate data that will be added to the memory, the fourth neural network—the candidate memory—is employed.

In practice, the LSTM unit uses recent past information (the short-term memory, H) and new information coming from the outside (the input vector, X) to update the long-term memory (cell state, C). Finally, it uses the long-term memory (the cell state, C) to update the short-term memory (the hidden state, H).

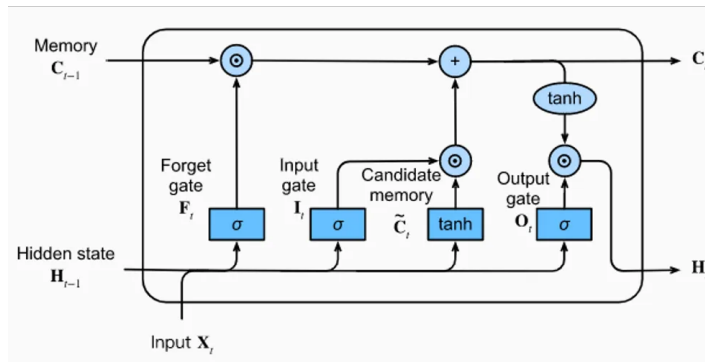


Figure 2.2: Recurrent Neural Network

The bidirectional[4] version involves duplicating the first LSTM layer in the network so that there are now two layers side-by-side, then providing the input sequence as-is as input to the first layer and providing a reversed copy of the input sequence to the second.

2.2.1 Structure of the network

The trained Bidirectional LSTM consists of:

- An *Embedding()* layer compressing the vocabulary size of 50000 to an embedding dimension of 32 for each word for a maximum of 14 words per sentence.
- A *Bidirectional()* layer with an output dimension of 128 for each sentence (64 for each of the two LSTM)
- A *Dense()* layer emdedding the output of the previous layer into a smaller dimension (16)
- A *Dropout()* layer to tackle overfitting A last *Dense()* layer with *sigmoid* activation function to return probabilities of belonging to the positive class (spam).

The model achieved an outstanding performance quickly converging to **100%** accuracy in training set and **98,5%** in the test set. The results are remarkable and entail that the model correctly classifies messages virtually all the times. Input data were straight-forward and a very good performance was indeed expected, as the task is not particularly difficult. However, given the almost perfect result achieved, we deemed opportune confronting the model performance in terms of accuracy to the one of an alternative model and check if the results are somewhat comparable.

2.3 BERT Transfer Learning

The alternative solution we applied to the spam classification problem was exploiting pre-trained models, in a practice called **transfer learning**. In particular, a recent breakthrough in the field of Text Mining led to the widespread employment of **transformers** networks such as **BERT**[5]. Researchers at Google AI Language recently published a paper called BERT that presents state-of-the-art findings in a range of NLP tasks, such as Question Answering (SQuAD v1.1), Natural Language Inference (MNLI), and others. The main technological advancement of BERT is the application of Transformer, a well-known attention model, to language modeling through bidirectional training. This is in contrast to earlier research that examined a text sequence either from the left to the right or by combining left-to-right and right-to-left training. An attention mechanism called *Transformer* is used by BERT to identify contextual relationships between words (or sub-words) in a text.

As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore it is considered bidirectional, though it would be more accurate to say that it's non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word).

Before delving into the modeling phase, we employed a tokenizer specific to BERT, the '*BertTokenizer()*'[6]. This package from the '*transformers*' library performs the following operations:

1. Encodes each word as a vocabulary index.
2. Adds special tokens [CLS] and [SEP] at the beginning and at the end of the sentence respectively.
3. Pads or truncates all observation to a fixed length.
4. Returns **attention masks**, which let the model know which tokens contain real

informations and which do not.

Many versions of BERT classifier models exist, proving how dominant this approach has become in the last few years. Nonetheless, we performed fine tuning with the former version '*bert-base-uncased*'[?] and 2 output classes (positive and negative) were set. Usually the lowest layer of the BERT model is fine tuned according to the specific task presented, but in this case we kept the original parameters as they constitute a better reference for the reliability of the performance of the *Bi-LSTM* network approach.

The model attained an accuracy of **86.84%** on the training set and **85.86%** on the test set. This result shows that the optimal performance of our first approach is indeed reliable, because although the accuracy dropped by more than 10% it still remains at a good level without fine-tuning the parameters to the specific task of spam detection. This suggests that the classification can be handled with relative ease by the proposed models and the difference in performance between the model is justifiable by the fact that the *Bi-LSTM* model's parameters were fine-tuned.

Chapter 3

Topic modeling

In addition to classifying messages as spam or not spam, we also sought to understand the underlying themes or 'topics' present in the spam messages. To do this, we employed a technique known as topic modeling, which is a powerful tool in the field of Natural Language Processing (NLP) that allows us to analyze, explore, and visualize text data in a meaningful way. It is a type of statistical model used for discovering the abstract 'topics' that occur in a collection of documents. Topic modeling is particularly valuable when dealing with short text segments, such as those found in web forums, comments, or reviews. In the context of our analysis, we applied LDA topic model to short text segments to address the challenges to understand the underlying themes or 'topics' present in the spam messages. This approach aims to develop an efficient and effective topic model for short text segments, and to investigate an unsupervised opinion spam framework that can be used to detect fake reviews and suspicious users, and identify products and businesses that have been targeted by the fake reviews.

The use of topic modeling in the context of spam detection allows for a more comprehensive analysis of the content and facilitates the identification of underlying themes or topics in the spam messages. This can provide valuable insights into the nature of the spam content and enhance the overall effectiveness of the spam detection system.

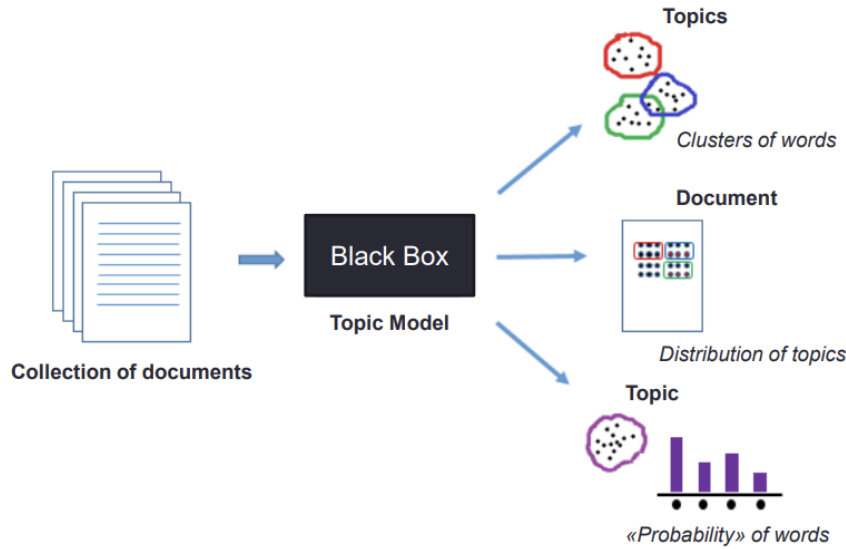


Figure 3.1: Topic modeling

3.1 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a widely used method for topic modeling in natural language processing. It is a generative probabilistic model that assumes each document is a mixture of topics, and each word in the document is attributable to one of the document's topics. In the context of our analysis, we applied LDA to our corpus of spam messages to identify the main topics. This involved preprocessing the text data, training the LDA model, and then analyzing the resulting topics. The LDA model was used from the Gensim library to perform topic modeling on a corpus of preprocessed documents. The model was trained with varying numbers of topics, and for each model, two key metrics were calculated for evaluation: coherence and perplexity.

- **Coherence:** Coherence measures the degree of semantic similarity between high scoring words in a topic. A higher coherence score indicates that the topic is more human interpretable.
- **Perplexity:** Perplexity is a measure of how well a probability distribution or probability model predicts a sample. Lower perplexity values indicate better models, as they suggest the model is more confident in its predictions.

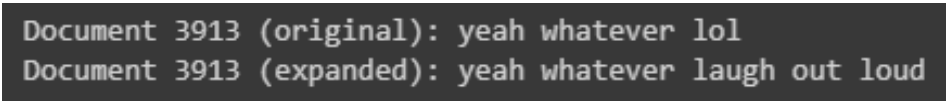
The LDA model was trained with different numbers of topics, and the resulting models were evaluated using these metrics to identify the optimal number of topics that best represent the underlying themes in the corpus of spam messages. The use of LDA for topic modeling provides a robust framework for uncovering the main topics in the corpus of spam messages, allowing for a deeper understanding of the content and facilitating more effective analysis and interpretation.

However, LDA also has some limitations and challenges that should be considered before using it. One of the main advantages of LDA is that it is an unsupervised learning technique, meaning that you do not need to provide any labels or categories for your documents. LDA can automatically infer the topics from the data, and assign each document a probability of belonging to each topic. This can save you time and effort. However, a disadvantage of LDA is that it can be computationally expensive and time-consuming, especially if the data is large, the number of topics is high, or the model is complex. LDA requires multiple iterations and optimization steps to estimate the topic distributions, which can take a lot of resources and memory. In conclusion, although LDA has some limitations, such as the computationally onerous topic inference phase, it continues to be a preferred choice in many situations. It was preferred to pLSA because it is more flexible and can adapt to new documents, while simply taking a sample from the Dirichlet distribution.

3.2 Abbreviation Expansion

Upon inspecting the dataset, a noticeable pattern emerged many email messages, particularly those classified as "ham", were rife with abbreviations. This abundance of abbreviations posed a potential challenge to accurate topic classification. To address this, a supplementary preprocessing step was introduced. This step involved the creation of a dictionary encompassing commonly used abbreviations in text messages, along with their corresponding full-word equivalents. The expansion of these abbreviations is a crucial preprocessing step in the analysis of SMS data, as it ensures that the text is represented in its full and unambiguous form. By expanding these abbreviations,

viations, the subsequent analysis can more accurately capture the semantic content of the messages, which is essential for the topic modeling task. The list covers a wide range of abbreviations, including those related to common expressions, emotions, and everyday language usage. This reflects the diverse and informal nature of SMS communication, highlighting the need to account for such linguistic characteristics in the analysis. Following the abbreviation expansion, we apply the expanded abbreviation list to all documents in the dataset. This ensures that the subsequent analysis is based on the expanded and unambiguous forms of the text.



```
Document 3913 (original): yeah whatever lol
Document 3913 (expanded): yeah whatever laugh out loud
```

Figure 3.2: Abbreviation Expansion example

The next crucial step involves splitting and preprocessing the expanded documents to prepare them for topic modeling. This process commences by constructing a Dictionary for the corpus and strategically filtering extremes to retain only pertinent and informative terms for subsequent analysis. Subsequently, a Bag of Words (BoW) representation is generated, forming the basis for the ensuing topic modeling phase.

In this comprehensive analysis, we adopt a meticulous approach by initializing variables to monitor the best model’s performance. To determine the optimal number of topics for our model, a list of candidate numbers of topics is defined (from a number of 2 topics to a number of 20 topics with a step of 2). Multiple iterations of the topic modeling process are then executed for each candidate number, with the calculation of both coherence and perplexity scores for each model. This is done for a more stable and reliable evaluation. The mean values of all of the iteration of each number of topic has been done. This practice helps mitigate the impact of potential randomness in the modeling process, ensuring that the results are more representative of the underlying structure of the data. The decision to conduct multiple iterations for each candidate number of topics is driven by the objective of reducing randomness in the modeling process. By averaging the scores over multiple runs, we aim to obtain a more robust and

consistent assessment of the model’s performance. This approach serves to counteract any fluctuations in the results that might arise from the inherent stochastic nature of topic modeling algorithms.

3.3 Evaluation

The primary objective of our analysis was to identify and interpret the predominant themes within the data, thereby providing valuable insights into the nature and composition of the messages. To ensure that the topics extracted from the model reflected genuine and distinct themes, we adopted a methodical and measured approach, employing various evaluation metrics and tools. These instruments not only facilitated the understanding and interpretation of the results but also ensured the reliability and validity of the analysis. Below, we discuss the main evaluation methodologies used in this study: Topic Coherence, Model Perplexity, and Topic Visualization.

In our investigation, we employed a Topic Modeling approach to identify and categorize the prevailing themes within a corpus of text documents. Utilizing a probabilistic method, we computed the likelihood of each document belonging to a specific topic. Subsequently, we assigned each document to the topic for which it exhibited the highest probability, thereby indicating the dominant theme for each individual document.

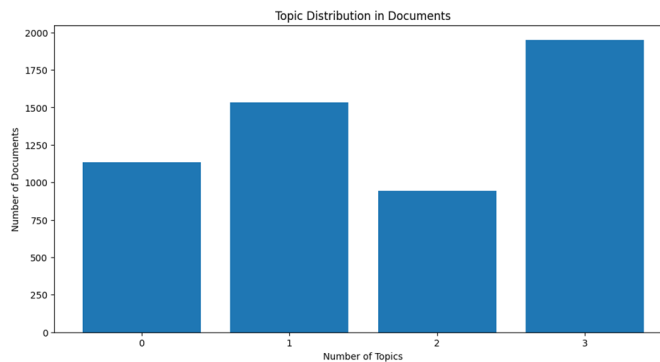


Figure 3.3: Topic Distribution

The bar chart showcased herein depicts the final distribution of documents among

the identified topics by the model. Each bar denotes a distinct topic, with the height of the bar indicating the number of documents most closely aligned with that topic. From the graph's analysis, it is evident that:

- Topic 0: Commands a moderate presence, suggesting a considerable number of documents are categorized under this theme.
- Topic 1: The second most dominant, actually reflects a lesser extent of document association compared to Topic 3.
- Topic 2: Holds the smallest number of associated documents, potentially reflecting a more specialized or infrequent theme within the corpus.
- Topic 3: Demonstrates the greatest number of associated documents, indicating that the theme it represents is highly prevalent in the dataset.

The significant representation of *Topic 3* suggests that the terms or phrases linked to this theme are widely utilized throughout the corpus, implying a theme of substantial interest or relevance to the dataset under review. The substantial presence of Topic 1, along with Topic 3, indicates that there are at least two prominent themes featured in the documents. The smaller number of documents assigned to Topic 2, although the least represented, could indicate a theme that, while not pervasive, may be critical to understanding the corpus, possibly signifying a distinct niche or a subset of particular interest.

3.3.1 Interactive Visualization of Topic Distribution and Saliency

During this phase of our textual data analysis, we employed the pyLDavis tool to create an interactive visualization that aids in the interpretation of Topic Modeling results. This tool integrates an intertopic distance map, showcasing the relationships between the various identified topics, with a list of the most relevant terms for each selected topic.

The visualization consists of two main components:

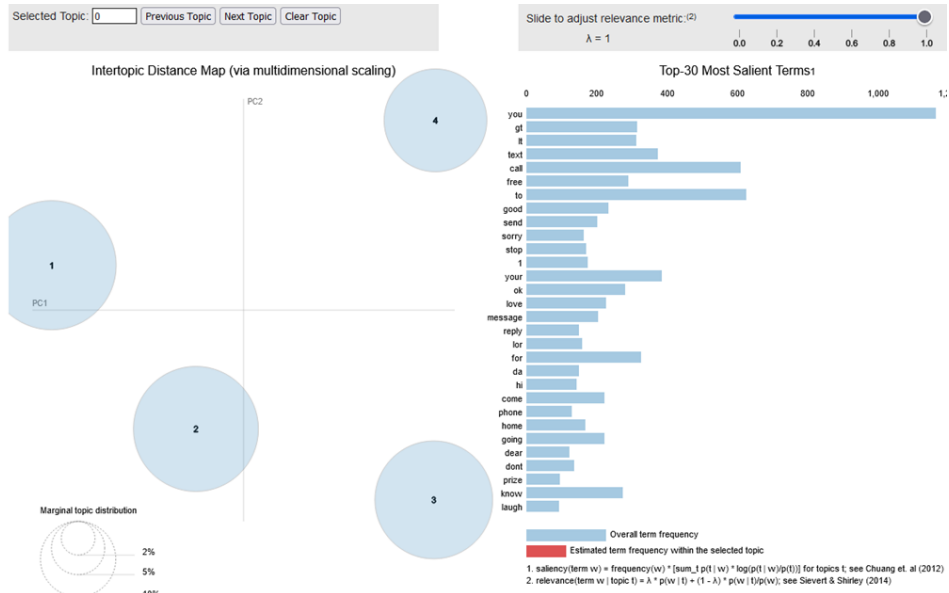


Figure 3.4: Topic Salience

1. Intertopic Distance Map: This is a bidimensional representation where topics are visualized as circles within a space defined by the first two principal components (PC1 and PC2). The size of each circle is proportional to the marginal topic distribution, representing the fraction of corpus tokens attributed to that topic.
 2. Most Salient Terms: This is a bar chart listing the 30 most salient terms for the selected topic. Salience merges the term frequency with its distinct importance to the topic, providing an indication of which terms are most characteristic and informative.
- Topic 1: Features terms related to personal or generic communications, with words such as "love," "home," and "dear." Its size suggests that it is a widely discussed theme in the dataset.
 - Topic 2: Includes words that may be linked to daily events or informal communications, with terms like "today," "hope," and "great."
 - Topic 3: Contains terms potentially associated with promotional messages or spam, such as "free," "prize," and "win."

- Topic 4: Demonstrates a prevalence of terms related to urgent or business communications, with words like "urgent," "call," and "contact."

pyLDA also allows the balancing of term visualization for a selected topic between those that are exclusively relevant to that topic and those that are the most frequent. The (λ) parameter controls this balance:

- When λ is set to 1, the bar chart displays terms ordered solely by their frequency within the selected topic. This may include terms that are frequent across the entire corpus and not just specific to the topic.
- As λ approaches 0, the bar chart prioritizes terms that are distinctive for the selected topic, even if they are not necessarily the most frequent. These terms are ones that better differentiate the topic from others.

The relevance metric used in pyLDAvis is defined as:

$$r(w|t) = \lambda \times p(w|t) + (1 - \lambda) \times p(w|t)/p(w)$$

Figure 3.5: Relevance Metric

- $p(w|t)$ is the probability of term w within topic t ,
- $p(w)$ is the probability of term w across the entire corpus.

This relevance metric helps find a balance between terms that are significant for a topic but might also appear frequently across other topics.

This interactive visualization allows us to effortlessly explore the data and understand how different topics unravel within the corpus. The spatial separation between topics on the map provides insight into their thematic distinction. For instance, topics that are distant on the map share fewer terms in common and represent more distinct themes. The size of the circles aids in comprehending the relative importance of each topic within the corpus.

Average Model Coherence: Model coherence measures the semantic similarity of the most frequent words within a topic. A high coherence score suggests that the keywords within a topic contribute to a cohesive and meaningful theme. We have calculated the average coherence across a range of models with varying numbers of topics to determine which number yields the most coherent and interpretable set.

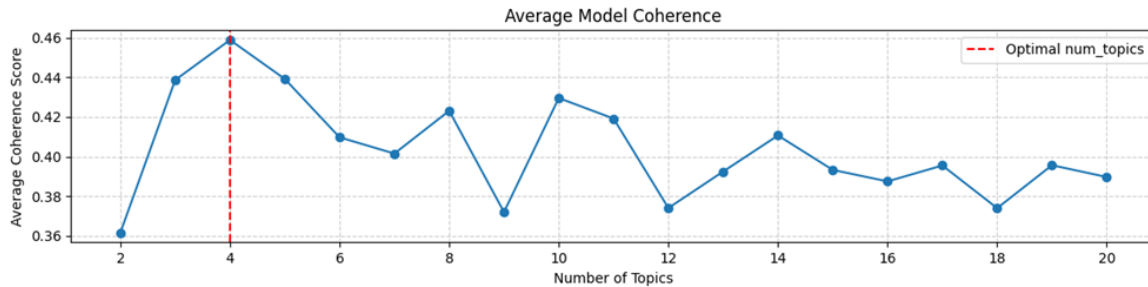


Figure 3.6: Average model Coherence

Each point on the graph corresponds to the average coherence score for a model with a specified number of topics. The graph reveals that coherence reaches its peak value when the number of topics is four (indicated by the dashed red vertical line). This suggests that a model with four topics is likely the most capable of capturing distinct and significant themes within the dataset. An average coherence score of 0.46 is relatively high in the context of Topic Modeling, indicating that the topics generated by the model consist of words that are significantly correlated with one another. This implies that within each topic, the keywords tend to contribute to a common theme or coherent discourse, making the topics readily interpretable.

Average Model Perplexity: Perplexity is a measure of how well a model predicts a sample. Generally, a lower perplexity score indicates a better-performing model. This metric was used to assess the ability of various Topic Modeling models to describe the corpus of documents.

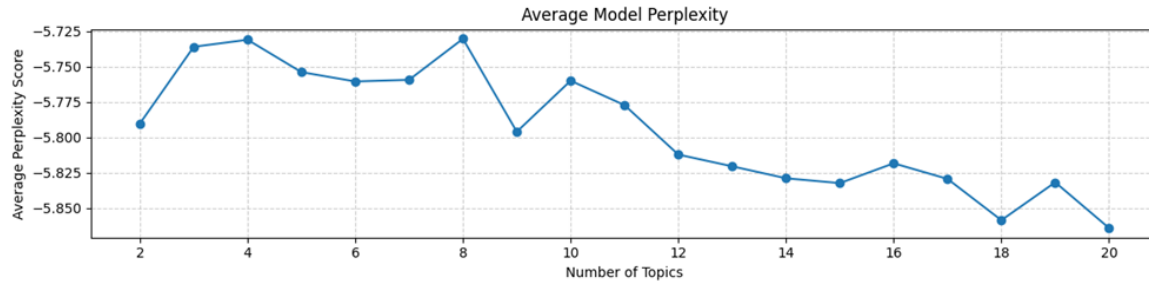


Figure 3.7: Average model Perplexity

The chart displays the average model perplexity as a function of the number of topics. Lower points on the graph indicate less perplexity and, hence, a more effective model. Perplexity generally decreases with an increasing number of topics, suggesting an improvement in the model’s predictive ability. However, an excessive number of topics might not necessarily translate into better qualitative interpretation of the data, as indicated by the coherence scores. Perplexity, denoted by a value of -5.73, is expressed as the logarithm of the actual perplexity, so the negative value does not indicate negative quality but is simply a consequence of the logarithmic transformation. In absolute terms, a lower (further from zero) logarithmic perplexity value implies better predictive power of the model. The combination of a relatively high coherence score and low perplexity indicates a high-quality Topic Modeling model. The model not only assigns high probabilities to the observed documents (as indicated by low perplexity) but also generates topics with highly correlated words (as indicated by high coherence).

Chapter 4

Conclusions

4.1 Classification

Spam detection proved to be a manageable task by the used models, confirming the effectiveness of *Bi-LSTM* and *Transformers* in Text Classification problems. Even without fine-tuning the trainable weights of the transformer on the specific task, a good performance was achieved. Further developments include gathering more noisy data to test the performance of the classifiers against harder tasks. Some potential improvements could be achieved by:

- Identifying specific tokens that are relevant to the spam detection task. These tokens could be removed from the stopwords list.
- Fine-tuning of the lowest layer of the transformer network by training the weights on the spam detection task.
- Transfer-learning the best large language models (**LLM**) in the state-of-the-art such as *LLama 2.0*, *GPT-4*, *Google Bard*, *etc.*, although this attempt is limited by computational requirements.

4.2 Topic Modeling

The Latent Dirichlet Allocation (LDA) algorithm in our Topic Modeling exercise has proven effective in identifying the main themes within our collection of spam messages.

Perplexity scores are crucial in assessing the performance of our topic model. Essentially, a lower score indicates that the model's predictions are more closely aligned with the actual distribution of words in the documents. In our study, a reduced perplexity score suggested that the model was proficient in forecasting the topics likely to emerge in new documents, signifying a strong correlation between the model and the data.

We also utilized pyLDAvis, an interactive visualization tool, which significantly enhanced our analysis. This advanced tool converted our topic modeling data into an interactive visual representation, enabling a clearer and more straightforward examination of the relationships among various topics. Through pyLDAvis, we could see how topics intersected, diverged, and grouped together, as well as gauge the prominence of each topic in the dataset. It also made it simpler to identify the key terms associated with each topic.

For future improvements to our Topic Modeling strategy, we are considering:

Updating our abbreviation expansion list by integrating user feedback and a more in-depth linguistic analysis to better reflect the evolving trends in SMS language. Experimenting with different numbers of topics beyond the currently identified optimal range, to potentially reveal more detailed sub-topics within the spam messages. Expanding our topic modeling to include temporal elements, allowing us to examine how the themes in spam messages may change over time due to shifts in communication styles or enforcement strategies.

Bibliography

- [1] “Detecting spam in mails.” <https://towardsdatascience.com/spam-detection-in-emails-de0398ea3b48>.
- [2] “Rnn.” https://it.wikipedia.org/wiki/Rete_neurale_ricorrenteD.
- [3] “Lstm.” <https://medium.com/@ottaviocalzone/an-intuitive-explanation-of-lstm-a035eb6ab4~:text=The%20LSTM%20unit%20is%20made,has%20four%20fully%20connected%20layers>.
- [4] “Bidirectional.” <https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/#:~:text=Bidirectional%20LSTMs%20are%20supported%20in,on%20to%20the%20next%20layer>.
- [5] “Bert.” <https://towardsdatascience.com/bert-explained-state-of-the-art-language-model->
- [6] “Bert guide.” <https://medium.com/@yashvardhanvs/classification-using-pre-trained-bert-model-transfer-learning-2d50f404ed4c>.