Context Aware Spam Detection

Student Name: Rathin Shah
Supervisor Name: Eamon Bell
Submitted as part of the degree of BSc Computer Science to the
Board of Examiners in the Department of Computer Sciences, Durham University

Abstract—The proliferation of spam comments on YouTube has become a significant challenge, negatively impacting user experience and undermining the platform's integrity. This research aims to explore the integration of contextual information into spam detection algorithms to enhance the accuracy and performance of these algorithms, while also leveraging state-of-the-art transformer-based models. By collecting a diverse dataset of YouTube comments and their associated contextual information, such as video titles, descriptions, tags, and categories, we explore various modeling approaches and techniques to improve spam detection performance. Our findings highlight the potential of the DistilBERT model in accurately classifying spam comments and demonstrate that incorporating contextual information can improve the model's ability to distinguish between spam and non-spam comments in certain scenarios. However, the extent of this improvement is relatively modest, and the specific combination of contextual elements does not substantially impact the model's performance. Furthermore, we develop a browser extension that integrates the trained spam detection model, showcasing the practical applicability of our research in providing users with a cleaner and more trustworthy YouTube comment section. This research contributes to the field of spam detection and content moderation by presenting a comprehensive approach that combines advanced natural language processing techniques with contextual information to combat the ever-growing problem of spam comments on YouTube.

Index Terms—Spam Detection, Browser Extension, YouTube Comments, Transformer models

1 Introduction

Online video platforms, notably YouTube, are essential hubs of content consumption, interaction, and feedback. With over 2.5 billion active users, YouTube is the second most used social media platform behind Facebook [1]. Millions of users flock to these platforms, not just to consume content, but also to voice opinions, share insights, and engage in discussions. The comment sections on YouTube videos serve as a vital space for users to express their thoughts, connect with others, and contribute to the overall discourse surrounding the video's content.

However, the quality and genuineness of these comment sections can often be compromised by the presence of spam comments. Spam comments can be described as those which have a promotional intent or those that are deemed to be contextually irrelevant for a given video [2]. These comments, whether generated by automated programs (bots) or humans, can significantly detract from the overall user experience by flooding the comment sections with promotional content, propaganda, or generally unengaging content [3].

The scale of the spam comment problem on YouTube is staggering. According to YouTube's transparency report, from October 2023 to December 2023, the platform removed a total of 1,197,687,868 comments for violating their Community Guidelines [4]. Remarkably, 99.6% of these removed comments were detected by automated flagging systems, while only 0.4% were detected through human flagging. Furthermore, an astonishing 83.7% of all removed comments were categorized as "Spam, misleading, and scams" [4]. These statistics underscore the immense challenge that YouTube faces in maintaining the integrity and quality of its comment sections, as well as the critical role that automated spam detection systems play in combating this issue.

The impact of spam comments extends beyond mere annoyance to users. They can hurt the reputation of a channel and undermine the trust and engagement of the community [2]. The presence of spam comments can make it challenging for users to find valuable and relevant discussions, leading to a deterioration in the quality of user interactions. Moreover, spam comments can also manipulate the perceived popularity or sentiment surrounding a video, potentially influencing the opinions and actions of viewers.

To address this growing concern, this research project aims to develop a spam detection system specifically designed for YouTube comments. The primary objective is to explore the impact of integrating contextual information, such as video titles, descriptions, and other metadata, into the spam detection process. In this research, 'context' refers to the additional information associated with a video, such as its title, description, tags, and category. These contextual elements provide valuable insights into the video's content, topic, and intended audience.

By considering the context of the comment, we hypothesize that the accuracy and effectiveness of spam detection can be significantly enhanced compared to traditional methods that rely solely on the content of the comments. The contextual information can help the spam detection system better understand the relevance and appropriateness of a comment concerning the video it is posted on. For example, a comment promoting a weight loss supplement on a video about cooking recipes would be considered spam, as it is not relevant to the video's context. By incorporating this contextual information into the spam

detection process, the model can better identify comments that are irrelevant to the video's content and classify them as spam more effectively. Thus, leveraging the contextual information can potentially improve the model's ability to distinguish between spam and legitimate comments.

While YouTube employs its own spam comment detection algorithm, the details of its inner workings are not publicly disclosed, as the platform understandably keeps its anti-spam measures confidential to prevent exploitation by spammers. According to YouTube's support documentation, their spam detection primarily relies on the text of a comment and the behaviour of the commenter, such as repeatedly posting similar comments [5]. This suggests that YouTube's current spam detection algorithm may not extensively utilize contextual information related to the video itself. However, without access to the actual algorithm, we cannot say with certainty to what extent, if any, contextual information is being used. As a result, researchers and developers seeking to improve spam detection techniques must rely on the available literature and explore novel approaches independently.

Upon reviewing the existing literature on spam detection, we found that the use of contextual information in spam detection algorithms has not been extensively explored, particularly in the context of YouTube comments. While many studies have focused on analysing the content of comments and user behaviour patterns, the incorporation of video-related metadata as contextual features has received limited attention.

Spam detection, at its core, is a classification problem. The goal is to classify each comment as either spam or non-spam based on its content. By treating spam detection as a classification task, we can apply various machine learning techniques, such as supervised learning algorithms, to learn patterns and characteristics that distinguish spam comments from genuine ones. The integration of contextual information into this classification process aims to enhance the accuracy and effectiveness of spam detection by providing insights into the relevance of a comment to the video, which can aid in identifying spam.

To achieve this, the research will involve collecting a dataset of YouTube comments, along with their associated video metadata. This dataset will be labelled, indicating whether each comment is spam or not. The labelled dataset will then be used to train and evaluate machine learning models that incorporate both the comment text and the contextual information. The performance of these context-aware models will be compared to traditional approaches that consider only the comment content, allowing us to assess the impact of incorporating contextual information on spam detection accuracy.

Furthermore, to demonstrate the practical applicability of this research, we will develop a browser extension that utilizes the trained spam detection models to filter out spam comments on YouTube in real-time. This extension will provide users with a cleaner and more trustworthy comment-viewing experience, showcasing the potential for significant improvements in online content moderation.

1.1 Project Deliverables

1.1.1 Basic

1) Comment Dataset - Find an appropriate dataset of spam YouTube comments and/or create our own dataset of spam and non-spam YouTube comments along with the video context information.
2) Basic Spam Detection - A baseline spam detection model that identifies spam based solely on the content of the comment and uses a Bag of Words model for text vectorization and a Machine learning model for classification.
3) Evaluation Framework - A method for evaluating the effectiveness of the spam detection algorithm, to be utilized at each stage of its development.

1.1.2 Intermediate

1) Contextual Spam Detection - Extract contextual information like video title, description, category and tags, and incorporate it into the baseline model.
2) Basic browser extension - A browser extension that can analyse comments and implement the spam detection algorithm, using pre-computed context data when available.

1.1.3 Advanced

1) State-of-the-art feature extraction - Instead of using Bag of Words, use transformers to vectorize the comment text and context.
2) Advanced Filter Extension - A browser extension that can extract contextual information directly from the browser to employ the advanced spam detection algorithm on any YouTube video, subsequently modifying the DOM to eliminate spam comments.

2 Related Work

2.1 YouTube comments analysis

The dynamics of user interactions on YouTube, particularly in the comment sections, are complex and multifaceted. Madden et al. [6] conducted a study that examined and categorised the various ways in which users made use of the YouTube comment section. A content analysis of 66,637 user comments on YouTube was done to create a classification schema that reveals ten broad categories and 58 subcategories. This is beneficial to understand the characteristics of both legitimate and spam comments.

Schultes et al. [7] conducted an in-depth analysis of 136,854 comments on 304 YouTube videos and classified these comments into 3 basic types:

1) T1 - ”Discussion posts” - comments part of a discussion among users
2) T2 - ”Inferior comments” - offensive statements, irrelevant content, spam
3) T3 - ”Substantial comments” - directly related to video content

These 3 types of comments had further subclassifications and analysis was done to understand the proportions of these classes across various types of videos and how the

proportions change based on the category of the video. For example, the videos under the category ”People and Blogs” had the highest percentage of the ”spam or offensive” comment type (28%).

It was found that the distribution of the ten comment classes is highly different among the video categories, and the variance can be explained mostly by the video content itself.

They also analysed how the proportion of each of the 3 basic types of comments listed above affected the dispersion of Likes and Dislikes on a video. It was found that an increased number of T2 comments (”inferior comments”) evokes a higher number of Dislikes. This highlights the importance of being able to effectively find spam comments that plague comment sections and deteriorate the user experience.

However, it is important to note that these papers are now quite old, and the type of spam comments, and consequently, their characteristics would have evolved to some extent.

2.2 Finding relevant YouTube comments

The concept of ”relevance” in the context of YouTube comments is crucial for understanding the value and impact of user-generated content on social media platforms, and eventually use ”relevance” as a factor in determining whether a comment is spam or not.

A. Marthe Möller et al. [8] define relevance as comments that reflect individuals' experiences of, or opinions or thoughts about the content of a video, or the artist/maker of the video. Additionally, comments that relate to the social experience of watching a video on a social media platform, such as referring to other users or asking for information, are also considered relevant. They also find that approximately 78% of YouTube comments under music videos are relevant, with most being relevant because they include a positive evaluation of the video, describe a viewer's personal experience related to the video, or express a sense of community among the video viewers.

They employed Supervised Machine Learning (SML) to automatically assess the relevance of comments written in response to music videos and concluded that SML is a suitable method to find those YouTube comments that are relevant to scholars studying viewers' reactions to online videos, and presented suggestions for scholars wanting to apply the same technique in their projects [8].

The importance of identifying relevant comments lies in the rich insights they can provide into viewers' reactions, opinions, and experiences of online videos. Social scientists often study comments on YouTube to learn about people's attitudes towards and experiences of online videos. However, the abundance of comments, many of which may be spam or off-topic, can pose a significant challenge to researchers. Irrelevant comments can require unnecessary processing and storage capacities and can seriously bias study findings [8].

2.3 YouTube Spam Comment Classifiers

The rapid growth of YouTube as a video sharing platform has attracted the attention of spammers who post irrelevant, misleading, or malicious comments to deceive users and promote their content. Spam comments not only degrade user experience but also pose security risks by spreading malware and phishing links. To combat this issue, researchers have explored various machine learning techniques for automated spam comment detection on YouTube.

Vimala Manohar Ruth et al. [9] conducted a comparative analysis of different machine learning algorithms for YouTube spam comment detection. They evaluated the performance of Logistic Regression, Decision Trees, Random Forest, AdaBoost, and SVM on a labelled dataset. Among the tested models, Logistic Regression achieved the highest accuracy of 95.4%, highlighting its suitability for this task.

Aiyar and Shetty [2] further extended their work by applying an N-Gram assisted approach for spam comment detection. They extracted frequently occurring character-level and word-level N-Grams from comments and used them as features for training Naive Bayes, Random Forest, and SVM classifiers. Their results showed that character-level N-Grams outperformed word-level N-Grams, with SVM achieving the highest F1-score of 0.984 using 6-grams. This study highlights the importance of capturing sequential patterns in comment text for effective spam detection.

While this paper provides valuable insights into the detection of spam comments on YouTube, it is important to note that there are potentially other types of spam that were not considered in this study. For instance, spam created simply to boost engagement or spam created to spread some sort of propaganda irrelevant to the video content. These additional types of spam warrant further investigation to fully understand the scope and impact of spam comments on online video platforms.

Alberto et al. [10] evaluated several top-performance classification techniques for filtering spam comments on YouTube, including decision trees, logistic regression, Bernoulli Naive Bayes, random forests, linear and Gaussian SVMs. The statistical analysis of the results indicated that these methods are statistically equivalent with a 99.9% confidence level.

Furthermore, the authors published five datasets that were used for their models [10]. These datasets were created by extracting comments from five of the ten most viewed YouTube videos during the collection period using the YouTube API. Each comment was manually labelled as spam or legitimate (ham) using a collaborative tagging tool that was developed for this purpose. This dataset can be valuable for our project as it provides a source of data that can be used to develop and test our spam detection techniques.

However, it is important to note that spam comments are always evolving to bypass whatever current techniques are being used to detect them. Given that the paper is eight years old, there will be new types of spam that also need to be accounted for.

In addition to their research findings, the authors developed an online tool called TubeSpam [10]. This tool

allows users to select a YouTube video, and it will then classify the comments of that video as ham or spam on the fly. This functionality is similar to the browser extension proposed in our project, although our browser extension would make it easier for users to filter spam comments because it does not involve going to a third-party site. Instead, the comments are filtered directly on the YouTube page itself. Furthermore, the TubeSpam website no longer seems to be available.

Hayoung Oh [11] proposed a cascaded ensemble machine learning model to enhance YouTube spam comment detection. They experimented with six different machine learning techniques, including Decision Tree, Logistic Regression, Bernoulli Naive Bayes, Random Forest, Linear SVM, and Gaussian SVM. By combining the outputs of these models using an ensemble approach with soft voting, they achieved superior performance across various evaluation metrics. While their work demonstrates the benefits of ensemble learning for spam detection, it does not explore the integration of contextual information from videos.

Meng Li et al. [12] focused on feature engineering for effective spam comment detection. They analysed spammer behaviour and comment content to derive two types of features - attribute features representing spammer characteristics and content features capturing semantic relevance. By combining these engineered features with a Gradient Boosting Tree classifier, they improved detection accuracy compared to previous methods. Their work emphasizes the importance of feature selection and extraction for spam detection.

Ramamonjisoa et al. [13] developed an online tool for automatically filtering YouTube comments based on five models trained on several datasets using deep learning algorithms, specifically utilizing BERT-Tiny, to classify YouTube comments. This model was fine-tuned to enhance performance in an online tool aimed at filtering comments in real-time. Their system not only classified comments as spam but also categorized them based on sentiment, irony, and constructiveness, demonstrating the diverse capabilities of machine learning models in understanding and moderating online discourse. This work aligns closely with the objectives of our project, emphasizing the use of advanced neural networks to improve the reliability and efficiency of spam detection.

Agarwal et al. [14] proposed the "DeepGram" algorithm, which combines deep learning-based language transformer models with N-gram-based machine learning models, for detecting spam comments on YouTube. The algorithm leverages the strengths of language transformers in capturing global context and semantic representations, along with the local patterns and features captured by N-gram-based models. The authors conducted experiments on a large dataset of YouTube comments and demonstrated that DeepGram achieves high accuracy and robust performance in detecting spam comments.

Abdullah et al. [15] conducted a comparative study of common filtering techniques used for YouTube comment spam, deploying datasets extracted from YouTube using its Data API. They evaluated the performance of nine different algorithms, including Adaptive Genetic Algorithm (AGA), Independent Component Analysis (ICA), and various machine learning techniques. The authors achieved high filtering accuracy (more than 98%) with low-complexity algorithms, suggesting the possibility of developing a browser extension to alleviate comment spam on YouTube.

The aforementioned studies have made significant contributions to YouTube spam comment detection using machine learning techniques. However, they primarily rely on comment text features and do not fully exploit the contextual information associated with videos. Our research aims to address this gap by investigating how incorporating video metadata, such as titles, descriptions, and other contextual elements, can enhance the accuracy and robustness of spam detection models.

2.4 Effect of advancement in AI and NLP

Advancements in AI and NLP technologies, such as large language models like ChatGPT, have enabled the generation of highly coherent and human-like text, making it increasingly difficult to distinguish between genuine comments and spam generated by sophisticated bots. Ferrara [16] discusses the novel challenges that have emerged in social bot detection due to these advancements, including the need for scalable and real-time detection methods, adversarial attacks, and ethical considerations.

The majority of existing literature focuses on identifying spam comments based solely on the content of the comment itself. However, with the advent of sophisticated Large Language Models and other AI-based natural language processing (NLP) systems that increasingly appear to "write" as effectively as humans, bots are now capable of generating text that is remarkably coherent and fluent, closely mimicking human-like text [17]. As a result, when a comment generated by such a model is examined in isolation, it becomes challenging to ascertain whether it is spam, given that the text is virtually indistinguishable from text generated by humans.

Our research aims to address these challenges by proposing a novel approach that incorporates video context, such as titles, descriptions, and metadata, to enhance the accuracy and robustness of spam detection models. By leveraging contextual information in addition to comment text, our method seeks to provide a more comprehensive and effective solution for identifying spam comments in the face of increasingly sophisticated AI-generated content.

3 Methodology

3.1 Data Collection

To construct a robust YouTube spam comment classifier that can also leverage contextual information, a comprehensive dataset comprising YouTube comments, labelled for spam, and the context of the corresponding videos is essential. However, the availability of datasets specifically curated for this task is notably scarce. Among the limited resources, one dataset stood out, containing 1,956 YouTube comments extracted from five videos that ranked among the top 10 in viewership during the collection period [19]. This dataset

consists of 1,956 real messages extracted from five videos that were among the 10 most viewed during the collection period. The dataset is divided into five sub-datasets, each corresponding to a specific YouTube video, as shown below in table 1. This dataset has been extensively used by other researchers working in the domain of YouTube spam comment detection [11] [10] [18].

TABLE 1
Description of existing dataset

Channel	YouTube ID	Spam	Non Spam
Psy	9bZkp7q19f0	175	175
KatyPerry	CevxZvSJLk8	175	175
LMFAO	KO6zrokCPj8	236	202
Eminem	ueHW8f7_U	245	203
Shakira	pRpeEdMmmQ0	174	196

To complement this existing dataset and increase the diversity of our data, we constructed our own dataset by selecting comments from three particular videos. For the first video, we retrieved the 1,000 newest comments. For the second and third videos, we aimed to retrieve the oldest comments to maximize the chances of finding spam comments. Since the YouTube API does not directly support retrieving the oldest comments, we had to employ a workaround. We first retrieved the newest comments, which YouTube provides in batches of 100 per "page", and traversed through all the pages until we reached the end. From there, we started collecting the comments in reverse order, effectively obtaining the oldest comments first. We obtained approximately 500 comments each for the second and third videos, resulting in a total of 2,021 comments across the three videos. Each comment within our dataset is assigned two labels: the first indicates whether the comment is spam, based on the content of the comment alone, and the second indicates its spam status in light of the contextual information.

To maintain focus on the video's content as the primary context, only top-level comments were collected. This decision is based on the rationale that the true context of a reply lies within the original top-level comment rather than the video content itself. Consequently, a reply may be irrelevant to the video but still not classified as spam if it pertains to the top-level comment it responds to. Furthermore, comments not written in English were labelled accordingly and filtered out before the data was used for training the models to ensure consistency and accuracy in spam detection.

No stopwords removal or any other preprocessing techniques are applied to the comments since research indicates that such techniques tend to hurt the performance of spam classifiers [21].

3.1.1 Spam Comment Criteria

The classification of comments as spam was guided by a set of predefined criteria, aimed at identifying various forms of irrelevant or manipulative content. These criteria include:

1) Promotion of unrelated channel/link/product.
2) Deceptive links / phishing.
3) Comments that have no clear relation to the video's content, including generic statements with an intention of promotion like: "Nice video, check out my channel" etc.
4) Comments that appear to have been generated by bots or automated tools, characterized by nonsensical text, random character strings, etc.
5) Comments that encourage participation in spammy activities like chain replying, purely for the sake of engagement on their comments.
6) Comments with very few number of words that are incomplete or meaningless sentences.

3.1.2 Criteria of video selection

In constructing the dataset, videos were meticulously selected based on a set of criteria designed to ensure the collection of a diverse and representative sample of comments. This included choosing videos with high view counts to guarantee a wide range of user interactions, spanning various content types to reflect different user communities, and focusing on those with active and recent comment sections to capture current spam trends. Additionally, videos with significant engagement and comprehensive contextual data were prioritized, all within the English language to maintain consistency in the dataset. Videos that garner more popularity tend to attract spammers, securing a broader audience in the process [31]. This pattern has also been observed in other online social platforms, such as Twitter [32].

3.1.3 Labelling process

Alongside the content of each comment, additional details were collected, including the comment ID, video ID, publication time of the comment, retrieval time, number of likes, and number of replies.

Once all the comments are obtained, a first pass is made over the comments where each comment is reviewed and labelled and spam or not solely based on the content of the comment and using the spam criteria mentioned above.

Following this initial classification, a second pass is made over these comments but this time the context of the comment is also included and considered when labelling the comment. By considering elements such as the video's title, description, and tags, this phase aimed to capture the nuanced ways in which context influences the perception and classification of comments as spam. It is important to note that while captions and thumbnails were gathered as part of the context information, they were not utilized while labelling the comments.

Furthermore, utilizing the Web Archive API through the *waybackpy* Python package, the web page of each video was archived. This step was taken to preserve the possibility of collecting additional contextual information in the future if needed.

3.1.4 Data Collected

Table 2 presents the number of spam and non-spam comments collected for each of the different YouTube videos.

3.2 Bag of Words + Naive Bayes

The baseline model for detecting spam comments is constructed using a Bag of Words (BoW) model for vectorizing the comments and context and a Multinomial Naive Bayes (MNB) classifier for categorization.

TABLE 2
YouTube comments collected for custom dataset

Video ID	Channel	Spam	Non spam	Total
uelHwf8o7_U	Eminem	82	918	1000
4r3TkdiMtcw	IShowSpeed	79	432	511
KOEfDvr4DcQ	MrBeast	50	460	510
External Dataset		1005	951	1956
Total		1216	2761	3977

The bag-of-words model is one of the most popular representation methods for object categorization [22]. It is a text representation technique where text is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity [23].

The Multinomial Naive Bayes classifier operates on Bayes' theorem, assuming independence between the features. For a document d, represented as a vector $\mathbf{x}$, the classifier predicts the probability $P(c|\mathbf{x})$ that d belongs to a class c, using the formula:

$$P(c|\mathbf{x}) \propto P(c) \prod_{i=1}^{N} P(x_i|c)$$

where $P(c)$ is the prior probability of class c, and $P(x_i|c)$ is the likelihood of feature i appearing in documents of class c [24].

The Bag of Words model is chosen for its simplicity and effectiveness in capturing the presence of words that are indicative of spam. The Multinomial Naive Bayes classifier is selected due to its efficiency and strong performance in text classification tasks, especially when dealing with discrete data like word counts. Moreover, it requires much less training data than most of the classifiers, and hence performs well on small datasets [25].

Initially, the dataset is loaded, and a preliminary filtering process is applied to remove non-English comments. Given the dataset's imbalance, with a predominance of non-spam comments, an undersampling technique is employed to equalize the distribution of spam and non-spam comments.

The Bag of Words and Multinomial Naive Bayes classifier are implemented through functions that are part of the *'scikit-learn'* Python module.

For feature extraction, the *'CountVectorizer'* function is utilized to implement a Bag of Words model. This method transforms the text data into a matrix of token counts, effectively converting the textual information into a format that can be processed by machine learning algorithms. Subsequently, the *'MultinomialNB'* function is employed to instantiate the Multinomial Naive Bayes classifier.

The dataset is split into training and testing sets with an 80-20 ratio. The model is then trained on the training set and evaluated on the test set, with the evaluation focusing on accuracy, precision, recall and f1-scores.

3.2.1 Incorporating context

To incorporate the contextual information of the comments into the training process, firstly the context information is loaded in using the video ID saved along with the comment. Initially, a basic approach is taken where all the context elements are simply concatenated together along with the main comment text to form one single feature. However, to emphasize the significance of the comment's content while still leveraging the contextual cues, the comment text is duplicated eight times before being combined with the title, description, tags, and category of the video.

The optimal number of times the comment text should be duplicated was found through trial and error.

3.3 Transformers + Logistic Regression

In this section, we present an advanced approach for spam comment classification that leverages state-of-the-art techniques in natural language processing (NLP). Building upon the baseline model of Bag of Words (BoW) and Naive Bayes, we explore the use of Transformers for text vectorization and Logistic Regression for classification. This method aims to capture the semantic and contextual information present in the comments and context and improve the overall performance of the spam detection system.

Transformers, specifically the BERT (Bidirectional Encoder Representations from Transformers) model, have revolutionized the field of NLP by providing powerful pre-trained language models that can effectively capture the meaning and context of text data. In our methodology, we employ the BERT model to vectorize the comment text and convert it into dense, high-dimensional embeddings.

BERT is a specific language model (LM) based on a large corpus of normalized versions of BooksCorpus and English Wikipedia. Language Models (Deep Learning Models) are playing an important role in Natural Language Processing such as GPT-3 [26].

BERT was chosen as the transformer model for this research due to its ability to capture bidirectional context in text, which is crucial for understanding the nuances and semantics of comments. Unlike traditional language models that process text sequentially, BERT considers the context from both directions, allowing it to better understand the relationships between words and phrases [26].

We begin by establishing a baseline model using a pre-trained model of BERT called 'bert-base-uncased' and considering only the comment text for classification, without incorporating any contextual information. The BERT-base-uncased model is pre-trained on a large corpus of text data using a self-supervised learning approach, allowing it to capture rich semantic and contextual information from the input text.

To vectorize the comments using BERT-base-uncased, we first tokenize the text using the 'BertTokenizer'. The 'BertTokenizer' is a component of the BERT model that converts the input text into a sequence of tokens. It applies various preprocessing steps, such as splitting the text into subwords (WordPiece tokenization), adding special tokens (e.g., [CLS] and [SEP]), and converting the tokens into numerical representations (token IDs).

The 'BertTokenizer' has a maximum sequence length of 512 tokens. If the input text exceeds this length, it is truncated to fit within the limit. On the other hand, if the input text is shorter than 512 tokens, padding tokens are added to ensure a consistent sequence length across all input samples. This padding is necessary because the BERT model expects a fixed-length input.

Once the text is tokenized, the resulting token IDs are passed through the pre-trained BERT-base-uncased model. The model processes the input sequence and generates embeddings for each token. These embeddings capture the semantic and contextual information of the tokens within the input sequence. In the baseline model, we extract the embedding corresponding to the [CLS] token, which is a special token added by the 'BertTokenizer' at the beginning of the input sequence. The [CLS] token embedding is typically used as a representation of the entire input sequence, as it is trained to capture the overall semantic information during the pre-training phase of BERT.

The extracted [CLS] token embeddings are then used as input features to train a Logistic Regression classifier. Logistic Regression learns a linear decision boundary in the feature space to separate the spam and non-spam comments based on the comment text embeddings.

3.3.1 Incorporating context

To enhance the performance of the spam comment classification model, we explore various approaches to integrate contextual information associated with each comment.

3.3.1.1 Concatenating Context with Comment: One approach to integrate contextual information is to concatenate the context with the comment text and treat the concatenated text as a single input to the Transformer model. We concatenate the video title, description, tags, and category with the comment text, forming a combined input sequence. The Transformer model then processes this combined sequence and generates embeddings for classification.

3.3.1.2 Using Context as a Separate Feature: Another approach is to treat the contextual information as a separate feature from the comment text. We concatenate the video title, description, tags, and category into a single context representation and vectorize it using the same Transformer model employed for comment vectorization. The resulting context embeddings are then used as additional features alongside the comment embeddings for classification.

3.3.1.3 Treating Each Context Element as a Separate Feature: We also investigate the impact of treating each context element (video title, description, tags, and category) as separate features. We vectorize each context element independently using the Transformer model and use the resulting embeddings as distinct features for classification.

3.3.1.4 Duplicating Comment to Increase Weighting: To investigate the importance of the comment text in relation to the contextual information, we experiment with duplicating the comment text multiple times before concatenating it with the context. The intuition behind this approach is to increase the weighting of the comment text in the input sequence and emphasize its relevance for classification.

3.3.1.5 Cosine Similarity between Comment and Context: We also explore the use of cosine similarity as a measure of semantic relatedness between the comment text and the contextual information. We calculate the cosine similarity between the comment embeddings and the context embeddings obtained from the Transformer model. The resulting similarity scores are then used as separate features for classification.

3.3.2 Experimenting with Different Transformers

To further enhance the performance of the spam comment classification model, we explore different types of Transformer models.

3.3.2.1 Sentence Transformers: Sentence Transformers are pre-trained models specifically designed for sentence-level tasks, such as semantic similarity and sentence embeddings. We experiment with the all-MiniLM-L12-v2 model, which is a lightweight and efficient variant of the Transformer architecture.

3.3.2.2 DistilBERT: DistilBERT is a distilled version of BERT that aims to reduce the model size while retaining most of the performance. It is trained using knowledge distillation, where a smaller model (DistilBERT) is trained to mimic the behaviour of a larger model (BERT).

3.3.3 Pooling Techniques

In addition to experimenting with different Transformer models, we also explore various pooling techniques for obtaining fixed-length representations of the comment text.

3.3.3.1 Extracting [CLS] Token: In the baseline model, we extract the embedding corresponding to the [CLS] token, which is a special token that represents the entire input sequence. This approach provides a straightforward way to obtain a fixed-length representation of the comment text.

3.3.3.2 Mean Pooling: Mean pooling involves taking the average of the embeddings of all the non-padding tokens in the input sequence. By averaging the token embeddings, we obtain a fixed-length representation that captures the overall semantic information of the comment text.

3.3.3.3 Max Pooling: Max pooling involves taking the maximum value along each dimension of the token embeddings. The intuition behind max pooling is to capture the most salient features or informative tokens in the input sequence.

3.4 Browser Extension

To demonstrate the practical application of our models, we developed a browser extension capable of extracting comments and contextual information from YouTube webpages and utilizing our Logistic Regression model to identify and filter out spam comments.

The architecture of the system is divided into two main components: the client-side browser extension and the server-side application. The client-side is implemented in JavaScript, leveraging web APIs such as fetch for asynchronous HTTP requests and DOM manipulation techniques to interface with YouTube's UI. On the server-side, Python is used with Flask to manage web server operations and flask-cors for handling cross-origin requests.

3.4.1 Client Side Implementation

Upon the user accessing a YouTube video, the extension script is triggered automatically. It begins by extracting the video ID from the URL's 'v' parameter.

The extension then employs the YouTube API along with the video ID to retrieve the contextual information of the video, particularly the video title, description, category, and tags. This information is subsequently sent to the server via

a POST request, allowing the server to store the video details for later use during spam classification.

Next, the extension starts periodically retrieving the comments from the YouTube webpage using the `document.querySelectorAll('ytd-comment-thread-renderer')` selector to extract all the comment elements on the page. For each comment element, the extension extracts the comment text and stores it in an array. It then sends the array of comment texts and the video ID to the server for spam classification.

Upon receiving the spam classification results from the server, the extension hides the comments that are classified as spam by modifying their CSS display property.

To handle the dynamic loading of comments, the extension periodically initiates the spam filtering process. It uses a timer to trigger the comment extraction and spam hiding process at regular intervals, ensuring that newly loaded comments are also processed and filtered for spam.

3.4.2 Server Side Implementation

The server-side application was implemented using Python with *Flask* [27], a lightweight and flexible web framework that provides a simple yet powerful foundation for building web applications. Flask's modular design and extensive ecosystem of extensions makes it well-suited for handling web server operations and managing HTTP requests [28].

Additionally, *flask-cors* [29] was used to handle cross-origin resource sharing (CORS) and ensure secure communication between the browser extension and the server. CORS is a security mechanism that restricts web pages from making requests to a different domain than the one that served the web page [30]. By properly configuring CORS headers, the server can allow specific origins or domains to make requests, preventing unauthorized access to server resources. This is particularly important in the context of browser extensions, where the extension script runs in the user's browser but communicates with a server hosted on a different domain.

When the server receives a POST request to the*/send_video_details* endpoint, it extracts the video details from the request body. It combines the title, description, tags, and category into a single context string. Next, the server vectorizes this context string using the DistilBERT model and stores this vector in memory for later retrieval based on the video ID.

When the server receives a POST request to the */predict* endpoint containing comment texts and the video ID, it retrieves the corresponding context vector from the stored dictionary using the video ID. The server converts the comment texts into vector representations using the same pre-trained BERT model used for context vectorization. It then concatenates the comment vectors with the context vector to create a combined vector representation for each comment.

The combined vectors are passed to the pre-trained Logistic Regression classifier for spam classification. The classifier predicts whether each comment is spam or not, returning a boolean array. The server sends the spam predictions back to the browser extension as a JSON response, indicating which comments are classified as spam.

4 RESULTS

In the evaluation of our spam comment classifier, we employ several standard metrics from the field of machine learning to assess the performance of our models comprehensively. These metrics include accuracy, precision, recall, and the F1 score.

4.1 Evaluation Metrics

4.1.1 Accuracy

Accuracy is the simplest and most intuitive performance measure, and it is simply the ratio of correctly predicted observations to the total observations. It is suitable for binary classification problems, such as our spam detection system.

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$$

4.1.2 Precision

Precision, also referred to as positive predictive value, measures the accuracy of positive predictions. Formulated as the ratio of true positives to the total predicted positives, precision is crucial in situations where the cost of a false positive is high. In the context of spam detection, a false positive would mean labelling a non-spam comment as spam, which would not be ideal as we could be removing valuable comments.

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

4.1.3 Recall

Recall, or sensitivity, measures the model's ability to identify all relevant instances (true positives) effectively. For spam detection, a high recall rate is crucial as it reduces the risk of spam comments going undetected.

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

4.1.4 F1 Score

The F1 Score is a harmonic mean of precision and recall, providing a balance between the two by taking both false positives and false negatives into account.

$$\text{F1} = 2 \times \left(\frac{\text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}\right)$$

4.2 Context vs No context

4.2.1 Bag-of-Words

The model that only used the content of the comments to identify spam obtained an F1-score of 0.87 while the model that also included context obtained an F1-score of 0.89.

4.2.2 Transformers

For the purpose of vectorizing the comment and context, various transformers were investigated. In the following experiments, all the context elements were combined into a single string and used as a separate feature from the comment in the Logistic Regression model.

The following are F1-scores obtained for the various transformers:

TABLE 3
Effect of including context for different transformers

Transformer model	Comment Only	Comment + Context
bert-base-uncased	**0.956**	0.951
bert-base-uncased + PCA	0.944	**0.946**
distilbert-base-uncased	0.956	**0.964**
all-MiniLM-L12-v2	0.923	**0.926**

PCA = Principal Component Analysis

The F1-scores in table 3 indicate that the inclusion of context in spam detection **does not consistently** enhance the model's performance. However, it's notable that the highest F1-score across all models was achieved when context was included, specifically with the distilbert-base-uncased model.

For one of the transformers, PCA was performed on the transformer embeddings to reduce the dimensionality of the embeddings and potentially give more weightage to other features like the cosine similarity score.

(Parameters used for PCA: *n_components=50, random_state=42*)

4.3 Incorporating Context in the Model

The various methods of incorporating context investigated are described below, and the corresponding results can be found in Table 4 (detailed descriptions given in Section 3.3.1):

Method 1: Comment and context are combined into a single string and used as one feature.

Method 2: All context elements are combined into a single string and used as an independent feature alongside comment.

Method 3: Each context element used as an individual feature alongside comment.

Method 4: Comment and context are combined into a single string and used as one feature, but comment is duplicated multiple times to increase the weighting of the comment.

Method 5: Cosine similarity between comment and context string used as an additional feature. All context is combined into a single string and used as a separate feature from the comment.

The most effective technique, as evidenced by the performance of the DistilBERT model in table 4, is Method 3, where each context element is treated as an individual feature alongside the comment text. This approach yielded the highest F1-score of 0.967, marginally outperforming Method 2, which involves combining all context elements into a single string and using it as an independent feature alongside the comment text.

It is apparent that DistilBERT consistently excels across almost all methods of context incorporation, highlighting its effectiveness and robustness in this type of task.

While Method 3 is marginally the best, the similarity in performance to Method 2 indicates that the consolidation of context elements does not significantly diminish the model's ability to discern spam. This could be beneficial from a computational efficiency standpoint, as treating the context as a single feature may reduce the model's complexity and, consequently, the processing time.

TABLE 4
Comparing various ways of including context

Incorporation Method	BERT	DistilBERT	Sentence Transformer
Combined with comment as single feature	**0.921**	0.908	0.908
All context combined into a single feature	0.951	**0.964**	0.926
Each context element as an individual feature	0.951	**0.967**	0.926
Single feature, comment duplicated multiple times	0.941	**0.949**	0.931
Cosine similarity between comment and context	0.951	**0.964**	0.924

While duplicating the comment proved to be beneficial for the Naive Bayes classifier, as demonstrated by the Method 4 scores, it worsened the performance of the classifier in this case. Furthermore, the utilization of cosine similarity scores (Method 5) did not significantly alter the performance from that of Method 2 or 3.

4.4 Which Context is Important?

To determine which contextual elements contribute most significantly to the spam classification performance, we conducted experiments comparing different combinations of context elements. Table 5 presents the F1-scores obtained for various combinations of the video title, description, category, and tags.

TABLE 5
Comparing different combinations of context elements

Context Elements	F1-score
Title + Description	0.964
Title + Description + Category	0.964
Title + Description + Tags	0.964
Description + Category + Tags	0.964
Title + Category + Tags	**0.966**
Title + Description + Category + Tags	0.964

For each of these pooling techniques, the transformer used was DistilBERT, and the method of incorporation was to combine all the context elements into a single string.

4.5 Pooling Techniques

To obtain a fixed-length representation of the comment text and context, we experimented with different pooling techniques commonly used in transformer-based models. Table 6 presents the F1-scores achieved by each pooling method (description of each method can be found in section 3.3.3.

For each of these pooling techniques, the transformer used was DistilBERT, and the method of incorporation was to combine all the context elements into a single string.

TABLE 6
Methods of extracting fixed-length embeddings

Technique	F1-Score
Extracting [CLS] token	0.949
Mean Pooling	**0.964**
Max Pooling	0.922

The highest F1-score was obtained with the Mean Pooling technique.

4.6 Browser Extension

The developed browser extension successfully integrates with the YouTube webpage, providing users with a seamless experience in filtering out spam comments. Upon installing the extension, users can browse YouTube videos as they normally would, without any additional configuration or setup required.

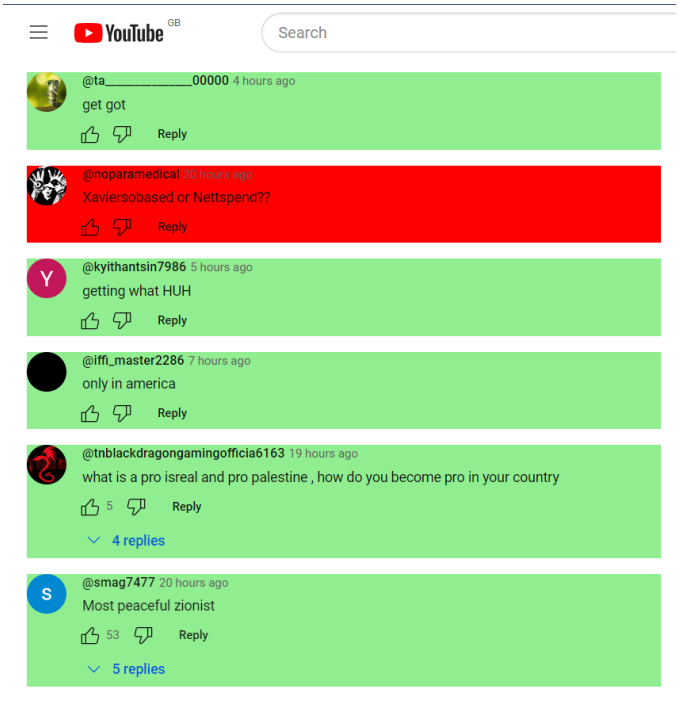

Fig. 1. Screenshot of extension higlighting spam comments in red and non-spam comments in green

As users navigate to a video page, the extension automatically obrain the video's contextual information through the YouTube API, such as the title, description, tags, and category, and send it to the server for preprocessing.

When the video page loads and displays the comments section, the extension starts extracting the comments visible on the page and sends them to the server for classification. The server, powered by the trained transformer-based model, analyzes each comment and determines whether it is likely to be spam or not.

The extension then receives the classification results from the server and dynamically updates the webpage, hiding the comments that are identified as spam. This process happens seamlessly in the background, without requiring any user intervention. From the user's perspective, the comments section appears cleaner and more focused, with spam comments effectively filtered out.

As the user scrolls through the comments or new comments are loaded dynamically, the extension continues its real-time scanning and classification process. This ensures that even newly added comments are promptly analyzed and filtered, maintaining a spam-free experience throughout the user's browsing session.

The extension's interface is designed to be unobtrusive and user-friendly. It does not require any manual input or configuration from the user, making it accessible to a wide range of YouTube users. The extension seamlessly integrates with YouTube's existing design and layout, ensuring a consistent and familiar user experience.

Figure 1 showcases the real-time spam comment detection capability of the browser extension. For the purpose of demonstration, the extension was configured to highlight non-spam comments with a green background and spam comments with a red background. This visual differentiation allows for a clear illustration of the extension's ability to accurately classify comments based on their content and the video's contextual information.

5 Evaluation

5.1 Context vs No Context

While using context improves the F1-score for some transformer models, the improvement in performance is marginal. Given this nuanced benefit, and considering that the inclusion of context can increase computational complexity and processing time, it may not always be justified to incorporate contextual data for marginal gains in performance.

The impact of contextual information on the model's performance is also influenced by the way in which it is incorporated. Experimenting with different approaches, such as treating the comment and each element of context as separate features or combining all context elements into a single string, provides insights into the optimal way to leverage contextual data. Using separate features for each context element achieved the best results but at the cost of significantly increased computation time and memory requirements, as each element needs to be individually tokenized and vectorized by the transformer. On the other hand, combining all context elements into a single string yielded comparable results with substantially lower computational overhead, making it a more efficient approach.

The choice of the F1-score as the primary evaluation metric for the spam comment detection model is justified due to its widespread use in binary classification tasks. The F1-score provides a balanced measure of a model's performance by considering both precision and recall. Precision measures the proportion of true positive predictions among all positive predictions, indicating the model's ability to

avoid false positives (i.e., incorrectly classifying non-spam comments as spam). Recall, on the other hand, measures the proportion of true positive predictions among all actual positive instances, indicating the model's ability to identify all genuine spam comments (i.e., minimizing false negatives). The F1-score, being the harmonic mean of precision and recall, offers a single metric that balances both aspects and is particularly useful when the class distribution is imbalanced, as is often the case with spam comments.

5.2 Which Context is Important?

Looking at the F1-scores in Table 5, we can see that while the combination of video title, category, and tags achieves the highest score of 0.966, the other combinations are not far behind, with all of them yielding an F1-score of 0.964. The difference between the highest and lowest scores is a mere 0.002, which is negligible in practical terms.

This small difference in F1-scores suggests that the specific combination of contextual elements used for spam comment classification might not have a significant impact on the model's performance. The model appears to be robust and able to effectively utilize various subsets of the available contextual information to distinguish between spam and non-spam comments.

One possible explanation for this observation is that the different contextual elements - title, description, category, and tags - may contain overlapping or redundant information. For example, the video title and description often cover similar topics, and the tags are typically related to the video's category. As a result, the model may be able to extract the necessary contextual cues from any combination of these elements, making it less sensitive to the specific elements used.

However, it is important to note that these findings are based on the specific dataset and the DistilBERT model used in this study. The generalizability of these results to other datasets, domains, or models may vary. It would be valuable to conduct further experiments with different datasets and models to confirm the robustness of these observations.

5.3 Data Collection

The process of creating a comprehensive and diverse dataset for spam comment classification presented several challenges. One of the primary difficulties encountered was the scarcity of spam comments available for collection. YouTube actively monitors and removes comments that violate their Community Guidelines, including those classified as spam. As a result, a significant portion of spam comments are deleted promptly, making it challenging to gather a large and representative sample of spam comments through conventional data collection methods.

Moreover, the manual process of collecting and labeling comments as spam or non-spam proved to be highly time-consuming. Each comment had to be carefully reviewed and assessed to ensure accurate labeling, which required significant human effort and attention to detail. The subjective nature of determining whether a comment constitutes spam further added to the complexity of the labeling process, necessitating the establishment of clear guidelines and consistency in annotation.

The limitations of the YouTube API also posed challenges in obtaining a comprehensive set of contextual information for each video. While the API provides access to video metadata such as the title, description, tags, and category, it does not offer straightforward methods to retrieve additional context elements like captions, chapter names, related video titles, or most replayed moments/timestamps. These additional contextual elements could potentially provide valuable insights for spam comment classification, but their absence limited the scope of the contextual information that could be incorporated into the dataset.

Despite these challenges, the decision to combine our manually collected dataset with an existing dataset proved beneficial in mitigating some of the limitations. Our dataset primarily consisted of spam comments that YouTube's filters had failed to detect, representing the types of spam comments that users are most likely to encounter. These comments are particularly important to consider, as they have evaded YouTube's existing detection mechanisms and require more advanced techniques for identification.

On the other hand, the existing dataset complemented our dataset by providing access to a wider variety of spam comments, including those that may have been swiftly removed by YouTube. This diversity in spam comments was crucial for training a more robust and generalizable spam classification model. By exposing the model to a broader range of spam patterns and characteristics, we aimed to enhance its ability to detect and classify spam comments accurately across different contexts and scenarios.

5.4 Spam Detection

The developed spam detection model and browser extension exhibit notable strengths, but it is crucial to acknowledge and address certain limitations. One significant challenge is the model's potential inability to effectively identify spam comments generated by sophisticated bots that are specifically designed to incorporate contextual information to bypass spam filters. These bots may employ advanced techniques to analyze the video's content, metadata, and other relevant contextual factors, crafting comments that appear relevant and genuine but are ultimately intended to promote spam content. As a result, the model may struggle to distinguish these carefully crafted spam comments from legitimate ones, leading to potential false negatives.

Additionally, the complexity of the transformer-based models used in the project introduces certain limitations. Transformers have a maximum token limit, which means that extremely long comments or video descriptions may be truncated during processing. However, in most cases, the entire comment or description may not be necessary to determine if a comment is spam, as the relevant information is often contained within a smaller subset of the text.

Despite these challenges, the use of transformer-based models, such as BERT and its variants (e.g., DistilBERT), has shown promising results in spam comment detection tasks. These models excel at capturing contextual information and semantic relationships within the text, making them well-suited for understanding the content and intent

of comments. The incorporation of contextual information, when done effectively, can provide additional insights and improve the model's ability to distinguish between spam and non-spam comments. However, the choice of the specific transformer model should be based on factors such as performance, computational efficiency, and the specific characteristics of the dataset and task at hand.

5.5 Browser Extension

The browser extension developed as part of this project serves as a valuable baseline implementation, showcasing the practical application of spam detection models in real-world scenarios. It demonstrates the potential for integrating advanced machine learning techniques into user-facing tools to enhance the browsing experience and combat the proliferation of spam comments. However, it is important to recognize that the current implementation has certain limitations that should be addressed in future iterations.

One notable limitation is the lack of user interaction features within the browser extension. In its current form, the extension does not provide users with the ability to manually report comments as spam or not spam. This functionality could be valuable in gathering user feedback and incorporating it into the model's training process, allowing for continuous improvement and refinement of the spam detection capabilities. User feedback can help identify false positives or false negatives, providing valuable insights into the model's performance and guiding future enhancements.

Furthermore, the current implementation of the browser extension employs a rather naive approach to handling dynamically loaded comments. As users scroll through the comment section, new comments are loaded incrementally, requiring the extension to process and classify these comments in real-time. The existing implementation simply sends all loaded comments to the server for classification every few seconds, regardless of whether the user has actively scrolled or interacted with the page. This approach can be inefficient and resource-intensive, especially for videos with a large number of comments.

To optimize the performance and efficiency of the browser extension, several improvements can be considered. For instance, the extension could be modified to only extract and process new comments when the user actively scrolls through the page, reducing unnecessary computations when the page is idle. Additionally, implementing a caching mechanism to store the classification results of previously processed comments can help avoid redundant API calls and improve response times.

6 Conclusion

This research project aimed to develop an advanced spam comment detection system for YouTube, leveraging state-of-the-art techniques in natural language processing, specifically transformer-based models, and incorporating contextual information to enhance the accuracy and effectiveness of spam identification.

The main findings of this research address the central question of whether integrating contextual information can enhance the performance of spam detection models for YouTube comments. The results demonstrate that incorporating context, such as video titles, descriptions, tags, and categories, can indeed improve the model's ability to distinguish between spam and non-spam comments in certain scenarios. However, the extent of this improvement is relatively modest, and the specific combination of contextual elements does not appear to have a substantial impact on the model's performance. Furthermore, given the marginal gains in performance and the potential increase in computational complexity and processing time associated with integrating context, the use of contextual information may not always be justified. This suggests that the decision to incorporate context should be based on the specific requirements and constraints of the system, such as available computational resources, real-time performance needs, and the acceptable level of performance improvement.

Furthermore, this study has explored different methods of incorporating contextual information into the spam detection process, revealing that treating each context element as a separate feature alongside the comment text yields the best performance. However, combining all context elements into a single string provides comparable results with significantly lower computational overhead, making it a more efficient approach in practical applications.

The development of a browser extension that integrates the trained spam detection model demonstrates the practical applicability of this research. The extension successfully filters out spam comments in real-time, providing users with a cleaner and more trustworthy YouTube comment section. This showcases the potential for integrating advanced machine learning techniques into user-facing tools to enhance the browsing experience and combat the proliferation of spam comments.

Despite the promising results, this study has also identified several challenges and limitations that warrant further investigation. The data collection process highlighted the difficulties in obtaining a large and diverse dataset of spam comments, as YouTube actively removes a significant portion of them. The manual labeling of comments proved to be time-consuming and subjective, emphasizing the need for more efficient and standardized annotation techniques.

Moreover, the developed spam detection model may struggle to identify spam comments generated by sophisticated bots that deliberately incorporate contextual information to bypass filters. As spam techniques continuously evolve, maintaining the effectiveness of the model over time remains a challenge. Future work should focus on developing adaptive and self-learning models that can keep pace with the ever-changing landscape of spam comments.

6.1 Future work: Browser Extension

The browser extension, while serving as a valuable baseline implementation, can be further enhanced by incorporating user interaction features, such as allowing users to report comments as spam or non-spam. This feedback loop can facilitate continuous improvement and refinement of

the spam detection capabilities. Additionally, optimizing the extension's performance by implementing efficient techniques for handling dynamically loaded comments and reducing unnecessary computations can greatly improve the user experience.

6.2 Future work: Fine-tuning Transformers

One significant future direction is to enhance the transformer-based models, such as DistilBERT, by fine-tuning them on domain-specific data. Although pre-trained models are effective for general natural language processing tasks, customizing them with a large dataset of YouTube comments will improve their contextual understanding and adaptation to platform-specific spam patterns. This fine-tuning approach ensures that models can better interpret the nuances of the YouTube commenting environment, increasing their ability to differentiate between genuine and spam content. Moreover, integrating cross-lingual fine-tuning will be beneficial, given the multilingual nature of the platform.

6.3 Future work: Using Thumbnails as context

Incorporating YouTube video thumbnails as an additional contextual feature represents another promising opportunity. Thumbnails often contain visual elements and textual overlays that can provide insights into the content type, theme, and audience of the video. For example, spam comments are likely to target specific video genres with greater frequency, which thumbnails can help identify. Utilizing computer vision models alongside natural language processing techniques could enrich the contextual information fed into spam classification models. By encoding the visual and textual data from thumbnails, the spam detection system can develop a deeper understanding of the relationships between video content and comment spam patterns.

REFERENCES

[1] Kemp, S. (2023). *Digital 2023 July Global Statshot Report.* 164. https://datareportal.com/reports/digital-2023-july-global-statshot

[2] Shreyas Aiyar and N. P. Shetty, "N-Gram Assisted Youtube Spam Comment Detection," Procedia computer science, vol. 132, pp. 174–182, Jan. 2018, doi: https://doi.org/10.1016/j.procs.2018.05.181.

[3] D. Bhavsar, S. Dcruz, A. Chandekar, M. Chaudhari, C. Patil, *A Research on YouTube Spam Comments Detection and Deletion,* vol. 11, no. 5, pp. 1108–1115, 2023, doi: https://doi.org/10.22214/ijraset.2023.51624.

[4] *Google Transparency Report,* Google.com, 2024. https://transparencyreport.google.com/youtube-policy/removals (accessed May 04, 2024).

[5] *Manage spam in comments - YouTube Help,* Google.com, 2019. https://support.google.com/youtube/answer/9482362 (accessed May 04, 2024).

[6] Madden, A., Ruthven, I., & McMenemy, D. (2013). *A classification scheme for content analyses of YouTube video comments*. Journal of Documentation, 69(5), 693-714. https://doi.org/10.1108/JD-06-2012-0078

[7] Schultes, Peter, Verena Dorner and Franz Lehner. *Leave a Comment! An In-Depth Analysis of User Comments on YouTube.* Wirtschaftsinformatik (2013).

[8] Möller, A. M., Vermeer, S. A. M., & Baumgartner, S. E. (2024). *Cutting Through the Comment Chaos: A Supervised Machine Learning Approach to Identifying Relevant YouTube Comments.* Social Science Computer Review, 42(1), 162-185. https://doi.org/10.1177/08944393231173895

[9] P. Vimala, M. Ruth, M. Khan, Y. Siva, and P. Reddy, *A Comparative Study on YouTube Spam Comment Detection Using Various Machine Learning Algorithms,* International Journal of Creative Research Thoughts, vol. 10, no. 6, pp. 2320–2882, 2022, Available: https://ijcrt.org/papers/IJCRT22A6718.pdf

[10] T. C. Alberto, J. V. Lochter and T. A. Almeida, *TubeSpam: Comment Spam Filtering on YouTube,* 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 2015, pp. 138-143, doi: 10.1109/ICMLA.2015.37.

[11] H. Oh, *A YouTube Spam Comments Detection Scheme Using Cascaded Ensemble Machine Learning Model*, in IEEE Access, vol. 9, pp. 144121-144128, 2021, doi: 10.1109/ACCESS.2021.3121508

[12] M. Li, B. Wu and Y. Wang, *Comment Spam Detection via Effective Features Combination,* ICC 2019 - 2019 IEEE International Conference on Communications (ICC), Shanghai, China, 2019, pp. 1-6, doi: 10.1109/ICC.2019.8761340.

[13] D. Ramamonjisoa, H. Ikuma and R. Murakami, *Filtering Relevant Comments in Social Media Using Deep Learning,* 2022 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), Niagara Falls, ON, Canada, 2022, pp. 335-340, doi: 10.1109/WI-IAT55865.2022.00056.

[14] A. Agarwal, P. Nikitha, S. Ramkumar, A. Sinha, P. Maheshwari, and A. S. Saini, *DeepGram: Combining Language Transformer and N-Gram based ML Models for YouTube Spam Comment Detection*, JDSIS, Nov. 2023

[15] A. O. Abdullah, M. A. Ali, M. Karabatak and A. Sengur, *A comparative analysis of common YouTube comment spam filtering techniques,* 2018 6th International Symposium on Digital Forensic and Security (ISDFS), Antalya, Turkey, 2018, pp. 1-5, doi: 10.1109/ISDFS.2018.8355315.

[16] Ferrara, E. (2023). *Social bot detection in the age of ChatGPT: Challenges and opportunities.* First Monday, 28(6). https://doi.org/10.5210/fm.v28i6.13185

[17] C. M. Anson, *AI-Based Text Generation and the Social Construction of "Fraudulent Authorship": A Revisitation*, Composition Studies, vol. 50, (1), pp. 37-46,179, 2022. Available: http://ezphost.dur.ac.uk/login?url=https://www.proquest.com/scholarly-journals/ai-based-text-generation-social-construction/docview/2693968025/se-2.

[18] N. M. Samsudin, C. F. binti Mohd Foozy, N. Alias, P. Shamala, N. F. Othman, and W. I. S. Wan Din, *Youtube spam detection framework using naïve bayes and logistic regression,* Indonesian Journal of Electrical Engineering and Computer Science, vol. 14, no. 3, p. 1508, Jun. 2019, doi: 10.11591/ijeecs.v14.i3.pp1508-1517.

[19] Alberto,T.C. and Lochter,J.V.. (2017). YouTube Spam Collection. UCI Machine Learning Repository. https://doi.org/10.24432/C58885.

[20] J. Depoix [jdepoix], "youtube-transcript-api GitHub Page," github.com. https://github.com/jdepoix/youtube-transcript-api/ (accessed Mar. 07, 2024).
[21] Almeida, T.A., Almeida, J. & Yamakami, A. *Spam filtering: how the dimensionality reduction affects the accuracy of Naive Bayes classifiers.* J Internet Serv Appl 1, 183–200 (2011).
[22] Zhang, Y., Jin, R. & Zhou, ZH. *Understanding bag-of-words model: a statistical framework.* Int. J. Mach. Learn. & Cyber. 1, 43–52 (2010).
[23] W. A. Qader, M. M. Ameen and B. I. Ahmed, *"An Overview of Bag of Words;Importance, Implementation, Applications, and Challenges,"* 2019 International Engineering Conference (IEC), Erbil, Iraq, 2019, pp. 200-204, doi: 10.1109/IEC47844.2019.8950616.
[24] C.D. Manning, P. Raghavan and H. Schuetze (2008). *Introduction to Information Retrieval*. Cambridge University Press, pp. 234-265.
[25] A. Raj, *"The not so naive Bayes - towards data science,"* Medium, Dec. 26, 2021. Accessed: Mar. 07, 2024. [Online]. Available: https://towardsdatascience.com/the-not-so-naive-bayes-b795eaa0f69b
[26] Brown, T.B., Mann, B., Ryder, N., Subbiah, M., et al. (2020) *Language Models are Few-Shot Learners.* [Online]. 2020. arXiv.org. Available at: https://arxiv.org/abs/2005.14165 (Accessed: 23 April 2024).
[27] *"Welcome to Flask — Flask Documentation (3.0.x),"* Palletsprojects.com, 2024. https://flask.palletsprojects.com/en/3.0.x/ (accessed Apr. 27, 2024).
[28] M. Grinberg, *Flask Web Development: Developing Web Applications with Python.* "O'Reilly Media, Inc."
[29] *Flask-CORS — Flask-Cors 3.0.10 documentation*, Readthedocs.io, 2023. https://flask-cors.readthedocs.io/en/latest/ (accessed Apr. 27, 2024).
[30] Mohammed Hussain Alrabrabah, *Understanding Access-Control-Allow-Origin (CORS) and How to Bypass it,* Medium, Nov. 09, 2023. https://medium.com/@moh.hussain06/understanding-access-control-allow-origin-cors-and-how-to-bypass-it-86533a97cd6b. (accessed Apr. 27, 2024).
[31] D. O'Callaghan, M. Harrigan, J. Carthy, and P. Cunningham, *Network Analysis of Recurring YouTube Spam Campaigns.* 2012.
[32] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, *Detecting Spammers on Twitter,* in Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS), Redmond, USA, 2010.