Reducing exposure to hateful speech online

Jack Bowker
Jacques Ophoff

Bowker, J. & Ophoff, J. (2022) 'Reducing exposure to hateful speech online'. In: K. Arai (ed.) *Intelligent Computing: Proceedings of the 2022 Computing Conference. vol. 3*. Springer, Cham, pp. 630-645, Computing Conference 2022, United Kingdom, 14-15 July 2022.

Reducing Exposure to Hateful Speech Online

Jack Bowker[1] and Jacques Ophoff[1,2][0000-0003-0634-5248]

[1] Abertay University, Dundee, United Kingdom
[2] University of Cape Town, Cape Town, South Africa
1803838@abertay.ac.uk, j.ophoff@abertay.ac.uk

Abstract. It has been observed that regular exposure to hateful content online can reduce levels of empathy in individuals, as well as affect the mental health of targeted groups. Research shows that a significant number of young people fall victim to hateful speech online. Unfortunately, such content is often poorly controlled by online platforms, leaving users to mitigate the problem by themselves. It's possible that Machine Learning and browser extensions could be used to identify hateful content and assist users in reducing their exposure to hate speech online. A proof-of-concept extension was developed for the Google Chrome web browser, using both a local word blocker and a cloud-based model, to explore how effective browser extensions could be in identifying and managing exposure to hateful speech online. The extension was evaluated by 124 participants regarding the usability and functionality of the extension, to gauge the feasibility of this approach. Users responded positively on the usability of the extension, as well as giving feedback regarding where the proof-of-concept could be improved. The research demonstrates the potential for a browser extension aimed at average users to reduce individuals' exposure to hateful speech online, using both word blocking and cloud-based Machine Learning techniques.

Keywords: Hate Speech, Machine Learning, Browser Extension.

1 Introduction

'Hate speech' is a form of targeted abuse, aimed towards an individual or group with the intent to offend or threaten [1, 2]. In online spaces, hate speech has long been an issue due to the anonymity and perceived lack of consequences of online speech, along with the ability for hateful groups to easily congregate.

It has been observed that exposure to hateful speech and sentiment towards marginalized groups can reduce levels of empathy in the wider population, with a Finnish survey reporting 67% of respondents aged between 15-18 years old had been exposed to hateful speech online, and 21% having fallen victim to such material [3]. The effect of hate speech on members of marginalized groups online is self-evident, as "From the perspective of members of the targeted groups, hate speech 'sets out to make the establishment and upholding of their dignity' much harder" [4].

Several different techniques have been attempted to reduce the amount of hateful speech online, with Google funding the 'Jigsaw' research wing, focused on fighting online toxicity using Machine Learning (ML). This process involves training a computer algorithm to learn and improve over time. The research involved gathering large datasets of comments and categorizing toxic sentiment, offering the ability to automatically block hateful comments from sites such as The New York Times for their comment section [5].

Even though most major social media companies have the data, resources, and experience to reduce the hate on their platforms using advanced techniques such as ML, the issue is politically sensitive, with the companies cautious to ban accounts or hide posts after accusations of political censorship [6]. In March 2019, CEO Mark Zuckerberg detailed Facebook's shift towards a more 'privacy-focused' platform, motivated by the public response to the Cambridge Analytica privacy scandal and spread of offensive content [7]. This move also reduces the scope of their responsibility to moderate content on the platform, with end-to-end encryption and private groups making moderation more difficult. These reasons are part of the reason why work is being done to give individuals control over what they see online, with Google's Jigsaw developing the experimental 'Tune' browser extension, allowing users to determine the intensity of speech they're exposed to.

The aim of this research is to investigate how a browser extension can be used to assist users in reducing their exposure to hate speech online, by researching existing solutions and surveying members of the public on a proof-of-concept prototype solution. Extensions can be an effective place to do this detection as they're relatively accessible to users, usually installed as an add-on through an online store to extend the functionality of the browser, as well as being positioned in the web browser where most social interactions on PCs are carried out. Extensions have been used to give users control over their exposure to hate speech, with open-source solutions such as the Negator tool [8] using a locally trained Natural Language Processing (NLP) model, a type of ML that can be used to detect hateful sentiment by considering the context of the comment and comparing it to the comments it has seen and been trained on in the past. This research is primarily focused on the acceptability and usability of such an approach.

This paper proceeds as follows. Section 2 presents a literature review into definitions of hate speech, as well as existing technologies available to manage hate speech. Machine learning, and how it can be used to detect sentiment, is also explored. Section 3 describes the development of a proof-of-concept browser extension and server. Section 4 describes the design of an evaluation survey that was conducted with participants from the public, as well as the results of the survey. Section 5 discusses and reflects on the design and technical aspects of the proof-of-concept extension. Lastly, Section 6 summarizes the research contribution and discusses limitations and suggestions for future work.

2 Background

This section will review how browser extensions as well as different technologies such as ML can assist in accurately detecting instances of hate speech online. It will also discuss how this effectiveness can be measured, how to notify users of this speech, and issues regarding building datasets. This area of research has been of significant interest recently due to the rapid improvement of ML and language recognition, as well as public discussion about censorship and how much or little social media companies should censor content on the platforms.

2.1 Defining Hate Speech

The term 'hate speech' does not have a universal definition, and the scope of the term can depend on which definition is used. The majority of developed democratic countries have laws defining and restricting the use of it to differing extents [9], with the United Kingdom defining Hate Speech as an expression of hatred towards someone on account of that person's "colour, race, disability, nationality, ethnic or national origin, religion, gender identity, or sexual orientation". Additionally, "Any communication which is threatening or abusive, and is intended to harass, alarm, or distress someone" is illegal as of the 1994 Criminal Justice and Public Order Act [2].

Although the United States doesn't have hate speech written into law due to Supreme Court rulings around the First Amendment [10], most major social media platforms define hate speech in similar terms as the United Kingdom. Facebook, the largest social media platform globally, defines hate speech as a direct attack on someone's protected characteristics, including "race, ethnicity, national origin, religious affiliation, sexual orientation, sex, gender or gender identity, or serious disabilities or diseases" [1].

It has been found that regular exposure to this content can be harmful to the individual or groups that are directly targeted. For example, the effects of anti-Semitic and homophobic hate speech can cause heightened stress, anxiety, depression, and desensitization [11]. This conclusion is supported by a study run by the Economist Intelligence Unit which finds that 65% of women experience hate speech online [12]. Unfortunately, it is difficult to define a profile for cyberhate targets, which can provide a starting point to identify at-risk individuals [13]. Therefore, any technology to manage hate speech needs to be broadly accessible and usable.

2.2 Technology to Manage Hate Speech

As online hate speech has moved to the forefront of public discussion, in part thanks to public campaigning and the growing unanimity of social media, tools have been developed to try and mitigate this. Research has been carried out in using techniques such as keyword detection, account blacklisting, and ML based approaches. There are numerous different proposed methods to alert the user of hateful content, ranging from blocking the post from view completely to just giving the user a notice that the account has a history of hateful conduct.

The Negator tool [8] makes use of NLP with an Aspect-based Sentiment Analysis (ABSA) model. The extension uses server-side processing to detect the intensity of the speech from 0 to 100%, taking into consideration the topic and who/what the speech is aimed at, then categorizing the speech into topics including "Abuse, Personal Attacks, Cyberbullying, Sexual Advances, Bigotry, Criminal Activity" and "Death Threats", and will block the post with a visual indicator notifying the user which category the speech falls under if the intensity is more than 60%. The tool takes a harsh stance on hiding hateful content and entirely hides posts that meet its own criteria with an interface notifying the user that the content has been blocked, giving the category the post falls into, with an option to view the post anyway.

The Shinigami Eyes tool [14] takes a different approach in notifying users of harmful content. The extension focuses on transphobic social media accounts and websites, with users submitting links as being 'anti-trans' or 'trans-friendly' that are manually reviewed before being added to the list, implemented using a bloom filter. Shinigami Eyes uses colour coded warnings marking hyperlinks and profiles as red (anti-trans) or green (trans-friendly). This approach is less harsh in that it still shows users the offending user's posts but can give extra context that might increase the likelihood of users not interacting with 'anti-trans' accounts. Users may submit reports via the extension interface by right clicking on a link to a profile or website.

Modha et al. [15] proposes a browser extension that implements the TRAC dataset, which classifies training data into sentiment categories of 'overtly aggressive', 'covertly aggressive', and 'non-aggressive', showing users the levels of each category embedded in the web page. The extension uses a similar method of colour coding as the Shinigami Eyes extension, with posts that pass a high threshold of confidence displayed completely in red, with medium displayed in yellow and non-hateful posts in green. Along with this, the levels detected by the model are shown directly above the comment as a number. The decision was justified in the paper by the distrust that the public has regarding algorithms and ML, partly due to the fact they are seen as a black box.

2.3 Machine Learning

Client-side and cloud models. As ML has gained popularity in business applications, commercial services are now common where users can rent computing power and take advantage of mature pre-existing models. Robust libraries and frameworks exist for a multitude of languages for building a model locally, with Python offering libraries such as scikit-learn, mlxtend and Shogun that can be configured with most ML algorithms involving transformation, decision trees, classification and more [16].

Datasets. A significant factor in training an accurate ML model is to ensure there is a well annotated dataset that is relevant to the topic being worked on. In the area of linguistics, the term 'corpus' refers to a collection of texts, especially regarding a particular subject. In the field of NLP, a corpus similarly refers to a structured set of

text or data, commonly used for training a model. However, classifying hate speech has proven to be a challenge, principally due to the loose definition of the term.

Ross et al. [17] discusses this in the context of training a dataset from the online discourse surrounding the European Refugee Crisis, with data gathered from 13,766 Tweets including a selection of offensive hashtags. The Tweets were categorized by annotators who were also given Twitter's official definition of hate speech, but the rate of agreement was still low with a Krippendorff's alpha of (α) = 0.38 (with 0 being complete disagreement and 1 being complete agreement). They conclude that there needs to be a stronger and more consistent definition of hate speech. Additionally, when annotating datasets, finding common characteristics of content users find hateful will be useful in building a more automated detection model.

MacAvaney et al. [18] focuses more broadly on the challenges and possible solutions of hate speech detection. An issue brought up is the data usage and distribution policies of major social media companies who want to restrict users scraping their platform for various reasons including legitimate user privacy concerns. This creates an issue on Facebook, the biggest social platform and location for a significant amount of online discourse, not allowing for scraping of content. Therefore, a useful stream of data is restricted – for example discussion in comments underneath a controversial news article. A related issue is the disproportionate language representation in hate speech datasets, with English making up the vast majority due to being the go-to language for online discussion.

Assessing accuracy. There are various methods to assess the accuracy of natural language models, and the appropriate measurement can vary depending on what the NLP model is intended to be used for. A reliable method available with certain datasets is making use of 'Dev set' data. This was implemented as part of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-1) where 130 teams helped annotate 15,000 Facebook comments and posts [19]. This large data pool meant a second non-overlapping corpus could be created, which came from similar sources to the intended training data making it a good testing environment. Once test data is acquired, a popular metric to describe the performance of ML models is a Confusion matrix [20].

2.4 Summary

The literature shows the potential applications a browser extension could have in the real world, with different methods effective at helping users control their exposure to hateful content and be informed of who they are interacting with. The papers reviewed include detailed discussion of the different natural language models for detection but conclude that each method has its drawbacks and can all perform well with high quality training data. The research also provided valuable insights into datasets, highlighting the value of properly sanitized and annotated data to help train and test language models.

3 Prototype Development

This section describes the development of a proof-of-concept browser extension and a Python application (nicknamed *HateBlocker*), with the browser extension used to scan webpages for elements, and a Python server to receive these elements and process them, sending back the results. The browser extension was developed for the Google Chrome browser. This decision was made as Chrome is currently the market leader in desktop web browsers. As browser extensions are developed primarily using cross platform web technologies like JavaScript, it would be trivial to port over to a non-Chromium browser such as Mozilla Firefox.

An iterative development approach was adopted. Originally the browser extension was designed to deal with both detecting and processing webpage data. However, during the development process it was found that this increases the complexity of the code significantly, also making it more difficult to swap out the processing algorithm at a later stage. To allow future changes to the processing algorithm, the decision was made to split the application into two distinct parts as shown in Fig. 1. A Chrome web extension parses the webpage elements and sends and receive this data, while a separate Python-based server retrieves the data, processes it, and sends it back to the web extension.

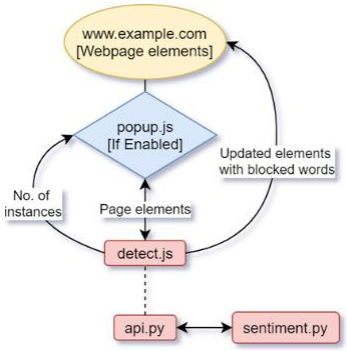

Fig. 1. Flow diagram of prototype

3.1 Text Detection

For the browser extension to detect and block hateful content on webpages, it needs to be able to read elements on the page, as well as filter which elements would be relevant to the search. To inject scripts into webpages, `content_scripts` must be declared through `manifest.json`. This file is required for the extension to run, as it holds the extension's metadata and other configuration. After fetching the relevant elements, the `innerText` of these elements was added to an array. This removed parts of the element object that were unnecessary for processing the text.

The technique used for fetching relevant elements depends on which site the *HateBlocker* is being aimed at. For the proof-of-concept the classic interface of the

popular Reddit social media platform was chosen (old.reddit.com). Reddit is a decentralized platform relying on voluntary moderators to manage communities. However, it's been widely reported that some communities don't strictly enforce Reddit's code of conduct and leave hateful content on the site [21], which makes it applicable to this research.

The classic interface was chosen as it doesn't rely extensively on JavaScript to draw interface elements, and therefore the Document Object Model (DOM) is relatively static and easy to parse. This method could easily be adapted to forums and imageboards such as 4chan. However, a different method would be needed for modern social media platforms such as Twitter, Facebook, and the most recent design of Reddit, as they use the popular ReactJS library. This library increases the complexity of detecting elements on a webpage because it virtualizes the DOM and only periodically updates the real version.

3.2 API

The development of the project was split into two parts, with the extension sending webpage elements via POST request as JSON data, a standardized format for serializing data, in real time to a remote server. This is done asynchronously using the fetch API available in JavaScript ES6. This data is then received by a Python Flask [22] application located at `app.py`. Flask was chosen as it's a lightweight web framework that allowed for a POST endpoint to be configured in a few lines of code. As shown in Fig. 2, two methods were added to the server, to receive POST requests and send this data to the analyser, and to send a response if a GET request is mistakenly sent to the server.

```
@app.route('/', methods=['POST'])
def post():
    a = Analyzer()
    data = json.loads(request.data)
    for sentence in data['data']:
        print(sentence)
        data['data'][sentence] = a.local(sentence)
    return json.dumps(data)

@app.route('/', methods=['GET'])
def get():
    return "Post your data to this endpoint for analysis!"
```

Fig. 2. Flask endpoints

Once data was passed into the Flask application and processed by the analyser, it was returned to the extension via the POST response, formatted as JSON data, with an extra property of either True, meaning the text was detected to be hateful, or False meaning the text didn't trigger the analyser. Fig. 3 illustrates how the endpoints communicate with the extension. This API based method of communication between the extension and server makes it easy for future work in to be carried out in implementing different extensions or servers.

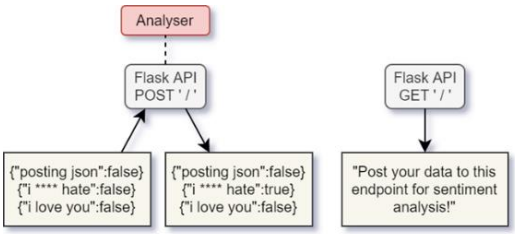

Fig. 3. API flow visualization

3.3 Text Processing

Text processing functionality was developed with the goal of being interchangeable, allowing for different hate speech detection techniques and algorithms to be added in the future. A local version of text processing was implemented in JavaScript in the Chrome extension earlier in the development stage. Different avenues were explored in this area, such as using a server-side wordlist and sentiment analysis using Google's Cloud Language and Perspective APIs.

Local wordlist. A local wordlist was the first step in detecting overtly hateful text online. This method misses out on the context in which the term was used - for example, if the word is offensive only in the context of a particular discussion. Although relatively easy to implement technically, a wordlist had to be found or generated for the Python application to use. Hatebase [23] provides a comprehensive and regularly updated list of hateful words in use in multiple languages globally and was considered for use in the application. The site requires use of an API to access their dataset which could considerably slow down the runtime of the application. An alternative dataset is made available by Carnegie Mellon University [24] listing 1,300+ English terms that could be found offensive. This dataset was downloaded as a TXT file and converted to CSV using Microsoft Excel, before being used in the prototype.

Cloud based sentiment analysis. After implementing word detection using a local Python server, cloud-based sentiment analysis was integrated into the application as a demo of the extensibility of a server-based approach. This was done using Google's Perspective AI. This is a limited access API developed by Jigsaw, a research unit within Google and Google's Counter Abuse Technology teams, to enable better conversations online by creating ML models targeted at online toxicity and harassment [25]. As shown in Fig. 4, the JSON response was parsed to get the toxicity score for the paragraph, measured between 0 (no toxicity) and 1 (extremely toxic), with any value over 0.5 returning True and resulting in the element being marked as hateful.

```
def perspective(self,text_content):
    # ...
    analyze_request = {
        'comment':{'text':text_content},
        'requestedAttributes':{'TOXICITY': {}}
    }
    if text_content:
        response = client.comments().analyze(body=analyze_request).execute()
        if response['attributeScores']['TOXICITY']['summaryScore']['value'] > 0.5:
            return True
    else:
        return False
```

Fig. 4. Perspective API implementation

3.4 Notifying the User

Due to how elements are sent to a server and returned, limitations were found regarding how the elements could be modified, to notify the user that they had been blocked. Initially, elements were checked to see if they included the blocked word, and if they did, the `innerHTML` was appended to remove the phrase and replace it with *"Post blocked by HateBlocker"*. The appearance of blocked elements is illustrated in Fig. 5. It uses a minimal notification, blending in with the page background. This method also allowed for users to view the content if they wished by selecting the dropdown (as shown in the bottom section of the figure).

Fig. 5. Appearance of blocked elements

As well as notifying the user by covering offending elements, the extension's pop-up was configured to show the number of instances on a given page. This was done by adding a temporary counter to Chrome's local storage API after instances are returned, which is fetched when the pop-up is opened. This result was cleared on page change, as well as when the extension was disabled using the pop-up interface.

4 Evaluation

A user acceptance survey was conducted to gather opinions on the proposed hate speech blocker browser extension. Due to COVID-19 restrictions, participants were unable to test the browser extension in person. Although options were considered with regards to allowing users to install the extension on their personal machines, the decision was made that this would be too awkward for novice users as well as being a security risk.

Due to the complicated nature of walking participants through installing a browser extension manually, it was decided that a video would be created to simulate what the browser extension's experience was like. This involved recording an in-development version of the extension on Reddit, a site the extension was intended for. The demonstration recording involved using the extension in a way that was as easy as possible to understand, which involved planning which areas of the extension to show and when. As shown in Fig. 6, a linear diagram was used to assist in visualizing the order to carry out the demonstration.

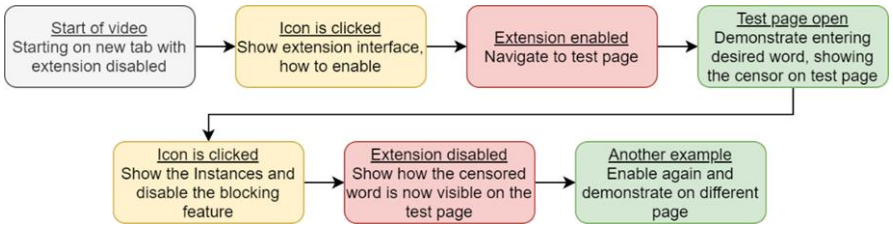

Fig. 6. Demonstration video visualization

Once the video was recorded, it was edited to slow down the clip and make it easier to follow. Annotations were added to explaining what was happening. The video was uploaded to YouTube from where it was embedded into the survey.

Participants were recruited using a combination of social media and email. A survey link was posted publicly to personal social media platforms (Facebook, Twitter, Instagram, and LinkedIn) and sharing was encouraged. The link was also sent out via email to personal contacts. Prior to data collection the survey was reviewed and approved by the School of Design and Informatics Ethics Committee at Abertay University.

4.1 Survey Items

Quantitative Items. This included demographic questions, as well as questions to assess the perceived usability of the extension. The System Usability Scale was used to assess participants' subjective rating on the usability of the extension. This scale was chosen as it's an established metric, proven to be highly robust and versatile [26]. It consists of 10 standard questions aimed at measuring the effectiveness, efficiency, and satisfaction of a system/piece of software, scored on a Likert scale from *Strongly Disagree* to *Strongly Agree*. These questions were adapted for the browser extension and are shown in Table 1. One additional Likert scale question was added, presenting the assertion that *"I'm comfortable with my amount of exposure to hateful content online"*. This question was added as a factor to determine how useful the extension would be for the user.

Table 1. System usability scale questions adapted for survey

Number	Question
Q1	I think I would use this extension frequently.
Q2	The extension looked unnecessarily complex.
Q3	The extension looked easy to use.
Q4	I think I'd need the support of a technical person to be able to use this extension.
Q5	I found the functionality of this extension well integrated with the site.
Q6	The extension was designed in an aesthetically pleasing way.
Q7	I would imagine that most people would learn to use this extension very quickly.
Q8	The extension looked very cumbersome to use.
Q9	I think I would feel very confident using the extension.
Q10	I would need to learn a lot of things before I could get going with this extension.

Qualitative Feedback. Three open-ended questions were added to gather more detailed perceptions. Firstly, *"Is there any other ways the browser extension could be improved, based on the video viewed?"* This question was used as general feedback for participants to voice their opinion and suggestions on how the extension itself should be executed. After this, the question *"Which websites would you see this extension being useful on?"* was asked to allow the user to add site suggestions they would find it useful on. Finally, the question *"If uncomfortable using this extension, what would make you more comfortable using an extension to limit your exposure to hate speech?"* was asked. This question is aimed at any participants that aren't comfortable with the concept of such an extension and gives an opportunity to explain what their reasons are.

4.2 Survey Results

In total 124 participants completed the survey. Slightly over half of participants (50.8%) were aged 18-24. Most participants identified as either male (44.4%) or female (42.7%). The highest level of education is University (50%), followed by higher/further education (30%), and a post-graduate degree (9.7%).

Participants were asked which social media platforms they use, to determine which platforms would be most useful to extend functionality to in the future. The major platforms identified were Facebook (76.6%), Instagram (71.8%), Twitter (71.8%), Snapchat (47.6%), Reddit (41.1%), and TikTok (27.4%).

Concerning the statement *"I'm comfortable with my amount of exposure to hateful content online"* (hereafter referred to as 'comfort with exposure') the participants generally held a neutral position ($M = 2.99$, $SD = 1.334$). Spearman's correlation was computed to assess the relationship between comfort with exposure and the demographic variables. There is a significant correlation between gender and comfort with exposure, with participants identifying as female and non-binary being less comfortable ($r = -.38$, 95% BCa CI [-.523, -.199], $p < .001$). This relationship is illustrated by the histogram in Fig. 7. This result supports prior research findings [12] and helps to identify at-risk individuals. No significant correlation was found between comfort with exposure and age, or level of education.

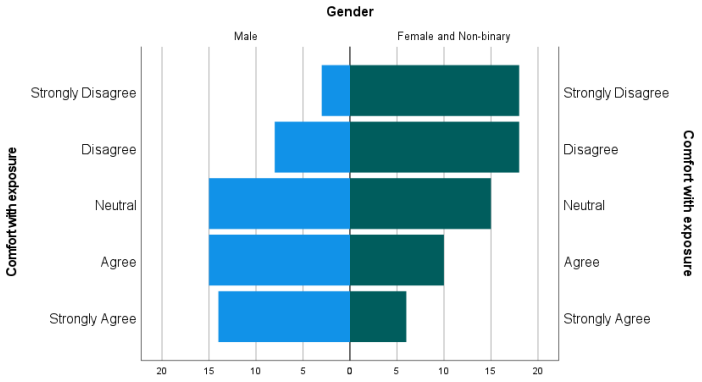

Fig. 7. Relationship between comfort with exposure and gender

The responses were analyzed to see which areas of the System Usability Scale stood out as strengths and weaknesses. The final score was found to be 77.3 out of 100, resulting in an above average score. Previous research has shown that scoring upwards of 71.4 would rank the result as 'good', with results of over 85.5 ranked as 'excellent' [27]. Fig. 8 shows the total responses for each question, keeping in mind that odd numbered questions are phrased positively (meaning *Strongly Agree* indicates better usability), and even numbered questions phrased negatively (and therefore *Strongly Disagree* indicating better usability).

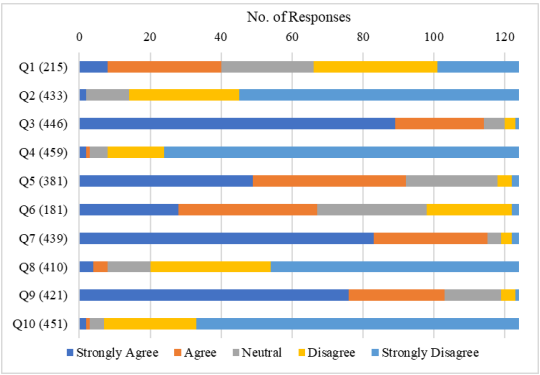

Fig. 8. Stacked bar graph showing System Usability Scale responses

Normalized scores for the questions are shown on the vertical axis, which takes account of the different phrasing and scores the questions on how positive the results were.

5 Discussion

5.1 Design

A high priority when designing the browser extension was ensuring it was accessible to users of all levels of technical proficiency. This was done by making the purpose of the extension clear through making the interface as simple as possible as well as providing materials to assist users in understanding how the extension works via a demonstration video.

The extension was developed through multiple iterations, with the first version demonstrated to survey participants in a Demonstration Video. The majority of the feedback regarding this version involved the fact that the text was too intrusive, as well as users being able to *"fill in the blanks"* of the blocked word since the rest of the sentence was visible, with over 30 responses mentioning this. For example: *"Perhaps find a way to block out sentences after the blocked word as you will know the word you have blocked out but an Excellent first step."* This feedback was addressed with the final version of the extension, as shown in Fig. 5.

In the initial design of the extension, it was proposed to show the user the specific type of hate speech the extension was hiding, but due to the method of word detection used, the extension could only tell whether an element was hateful or not, and not the specific reason it was. Therefore, the number of instances on the page were displayed to inform the user. This functionality could be implemented in the future by using a local ML model. The decision was made not to include a local ML model for the proof-of-concept due to the Perspective API being perceived as more accurate and being easier to implement, but after testing was found to be unusably slow when used with complex webpages. Sentiment analysis could be integrated to achieve a similar level of responsiveness as word-detection if implemented locally, while also opening the possibility of giving context-dependent warnings.

Survey participants praised the overall ease of use of the extension, with Q2, Q3, and Q4 related to the level of knowledge required to use it, with Q4. *"I think I'd need the support of a technical person to be able to use this extension"* being the highest scoring answer with an average response of 1.2 and the majority of participants choosing *Strongly Disagree* which is a sign that users understand the purpose and usage of the extension, leaving responses such as: *"It looks clean and easy to use as is."*

5.2 Technical Implementation

A core objective of this research was to develop a browser extension that would be able to detect and block hateful content on webpages. The extension was successful in this regard in that it effectively detected relevant elements of the target page, sending these using an API to communicate with the server that processed these results. It was also used to inform users of the status of the extension, and the number of instances of hate speech on any page.

A common point of feedback found in survey results regarding the technical functionality of the extension had to do with integrating a 'wordlist' or 'machine learning'. For example, *"I think the proposed method of adding words to a block list or using machine learning would be a good method to block words however I would like to see a user manageable list as some people have different qualifiers as what is hate to them."* This functionality (a wordlist and cloud-based sentiment analysis powered by the Perspective API) was implemented in the final version of the extension.

A targeted approach was used when building the detection functionality of this extension, and this significantly simplified the process of detecting elements on the chosen site, where the Paragraph tag was used. A downside of this method of element detection was that a lot of irrelevant elements were captured and processed such as page headings and navigation/side bars. This didn't cause issues when carrying out local word detection, as the API traffic stayed on the local machine, however if this server was moved to a remote location, it's possible the network connection would be a bottleneck.

Along with this extension, a server was implemented to carry out the processing function of the browser extension, enabling the two systems to exist independently. This would allow the browser extension to be modified and configured with different detection models, and the server to be used with any application that wishes to integrate with the HTTP API. Python was the language of choice for this server, due both to the simplicity of configuring an API using Flask and the wide range of libraries available for text processing and Machine Learning. The extension was built with an intent to make the process of integrating new detection algorithms simple, with a standard JSON input format of `sentence:true/false` and each method contained within a function inside the analyser. The server-side approach was successful in that it integrated well with the extension and performed well when working with a local word detection model.

Problems were experienced however when working with the remote Perspective API, where due to every paragraph element of the page being individually processed, often the rate limit of 60 requests per minute was reached before classifying all the elements on the page. Although the same rate limits were not experienced with the similar Google Cloud Language, this API was less catered towards toxicity and similarly to Perspective API, the time spent waiting for a response made these methods unusably slow on websites that were more complex.

6 Conclusion

The aim of this research was to investigate how a browser extension can be used to reduce exposure to hate speech online. It found, through investigation of existing literature, that NLP can play a valuable role in hate speech detection due to its ability to recognize the context in speech, and how browser extensions can be an effective method of managing what users are exposed to online. Through the development and evaluation of a proof-of-concept extension, it was found that users were receptive to this method of reducing their exposure to hate speech online.

This research was mainly focused on the acceptability and usability of the approach for users. It was found that the extension was highly usable, with an overall System Usability Score of 77.3 (good). In addition, survey results suggest that the public are open to using browser extensions or similar solutions to reduce their exposure to hate speech online, with most of the negative feedback related to the fact that the demonstration was based on an early version of the extension. Due to COVID-19 restrictions, the extension could only be demonstrated to participants using a video recording of its functionality. Although it is believed that this still presents a sufficient basis for evaluation, additional in-person evaluations would be beneficial.

It was found that sentiment analysis and NLP, in general, could greatly assist in reducing the amount of hateful speech online, especially when online platforms integrate it to deal with toxic accounts directly. Browser extensions, however, have the potential to play an important role for individuals that wish to cater their online experience separate from what online platforms deem as acceptable. The survey results indicate that gender could be an important indicator of at-risk individuals, with those identifying as female and non-binary being significantly less comfortable with the amount of exposure to hateful content online. Future research could investigate the impact on these groups in more detail, and what they would prioritize in a technical solution.

The extension was designed to be simple for the end user, whilst maintaining a level of interoperability and extensibility to add functionality in the future. For this purpose, it was designed as two parts, and using a standard API allows for the server or extension to be modified without having to make major changes to the protocol. While the proof-of-concept only targeted the classic interface of Reddit, the survey results showed this to be one of the top five platforms in use and thus relevant in the context of this research. The survey results confirmed several other platforms for future targeting.

A limitation present in the current version of the extension was the choice made between having a less accurate and context unaware word detection model that processes quickly, or a more accurate context aware sentiment analysis model that hinders performance. This limitation could be addressed by using a local ML model. However, a larger challenge would be bringing this technology to increasingly popular mobile platforms where content injection is impossible in most instances. How best to manage these constraints presents an opportunity for further research.

References

1. Facebook: Hate speech | Transparency Centre, https://transparency.fb.com/en-gb/policies/community-standards/hate-speech/, last accessed 2021/11/15.
2. The Crown Prosecution Service: Hate crime | The Crown Prosecution Service, https://www.cps.gov.uk/crime-info/hate-crime, last accessed 2021/11/15.
3. Oksanen, A., Hawdon, J., Holkeri, E., Näsi, M., Räsänen, P.: Exposure to Online Hate among Young Social Media Users. In: Soul of Society: A Focus on the Lives of Children & Youth. pp. 253–273. Emerald Group Publishing Limited (2014). https://doi.org/10.1108/S1537-466120140000018021.

4. Barendt, E.: What Is the Harm of Hate Speech? Ethic Theory Moral Prac. 22, 539–553 (2019). https://doi.org/10.1007/s10677-019-10002-0.
5. Google: Toxicity, https://jigsaw.google.com/the-current/toxicity/, last accessed 2021/11/15.
6. Gogarty, K., Silva, S.: A new study finds that Facebook is not censoring conservatives despite their repeated attacks, https://www.mediamatters.org/facebook/new-study-finds-facebook-not-censoring-conservatives-despite-their-repeated-attacks, last accessed 2021/11/15.
7. BBC News: Zuckerberg outlines plan for "privacy-focused" Facebook, https://www.bbc.com/news/world-us-canada-47477677, (2019).
8. Jain, S., Kamthania, D.: Hate Speech Detector: Negator. Social Science Research Network, Rochester, NY (2020). https://doi.org/10.2139/ssrn.3563563.
9. Howard, J.W.: Free Speech and Hate Speech. Annual Review of Political Science. 22, 93–109 (2019). https://doi.org/10.1146/annurev-polisci-051517-012343.
10. Harvard Law Review: Matal v. Tam Leading Case: 137 S. Ct. 1744, https://harvardlawreview.org/2017/11/matal-v-tam/, last accessed 2021/11/15.
11. Leets, L.: Experiencing Hate Speech: Perceptions and Responses to Anti-Semitism and Antigay Speech. Journal of Social Issues. 58, 341–361 (2002). https://doi.org/10.1111/1540-4560.00264.
12. The Economist: Measuring the prevalence of online violence against women, https://onlineviolencewomen.eiu.com/, last accessed 2021/11/15.
13. Manqola, T., Garbutt, M., Ophoff, J.: Cyberhate: Profiling of Potential Targets. CONF-IRM 2020 Proceedings. (2020).
14. Shinigami Eyes: An extension that highlights trans-friendly and anti-trans social network pages, https://shinigami-eyes.github.io/, last accessed 2021/11/15.
15. Modha, S., Majumder, P., Mandl, T., Mandalia, C.: Detecting and visualizing hate speech in social media: A cyber Watchdog for surveillance. Expert Systems with Applications. 161, 113725 (2020). https://doi.org/10.1016/j.eswa.2020.113725.
16. Stančin, I., Jović, A.: An overview and comparison of free Python libraries for data mining and big data analysis. In: 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). pp. 977–982 (2019). https://doi.org/10.23919/MIPRO.2019.8757088.
17. Roß, B., Rist, M., Carbonell, G., Cabrera, B., Kurowsky, N., Wojatzki, M.: Measuring the Reliability of Hate Speech Annotations: the Case of the European Refugee Crisis. Presented at the NLP4CMC III: 3rdWorkshop on Natural Language Processing for Computer-Mediated Communication 22 September 2016 September (2016). https://doi.org/10.17185/duepublico/42132.
18. MacAvaney, S., Yao, H.-R., Yang, E., Russell, K., Goharian, N., Frieder, O.: Hate speech detection: Challenges and solutions. PLOS ONE. 14, e0221152 (2019). https://doi.org/10.1371/journal.pone.0221152.
19. Kumar, R., Ojha, A.Kr., Malmasi, S., Zampieri, M.: Benchmarking Aggression Identification in Social Media. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018). pp. 1–11. Association for Computational Linguistics, Santa Fe, New Mexico, USA (2018).
20. Malmasi, S., Zampieri, M.: Detecting Hate Speech in Social Media. arXiv:1712.06427 [cs]. (2017).
21. Newcomer, E., Bloomberg: Racism is rampant on Reddit, and its editors are in open revolt, https://fortune.com/2020/06/20/racism-reddit-editors-in-revolt/, last accessed 2022/01/24.

22. Flask: Welcome to Flask — Flask Documentation (1.1.x), https://flask.palletsprojects.com/en/1.1.x/, last accessed 2021/11/15.
23. Hatebase: Hatebase, https://hatebase.org/, last accessed 2021/11/15.
24. von Ahn, L.: Offensive/Profane Word List, https://www.cs.cmu.edu/~biglou/resources/, last accessed 2021/11/15.
25. Perspective: About the API - FAQs, https://support.perspectiveapi.com/s/about-the-api-faqs, last accessed 2021/11/15.
26. Bangor, A., Kortum, P.T., Miller, J.T.: An Empirical Evaluation of the System Usability Scale. International Journal of Human–Computer Interaction. 24, 574–594 (2008). https://doi.org/10.1080/10447310802205776.
27. Bangor, A., Kortum, P., Miller, J.: Determining what individual SUS scores mean: adding an adjective rating scale. J. Usability Studies. 4, 114–123 (2009).