Human-in-the-Loop Machine Learning for Real time Feedback Systems.

Anish Kamble
Department of Mathematics and Statistics
University of Limerick

Supervisor
Dr. Eberhard Mayerhofer

MSc Thesis
MSc in Data Science and Statistical Learning
August 15, 2025

Abstract

This paper will present a Human-in-the-Loop (HIL) machine learning system that would enhance the accuracy, ethicality, and situational awareness of detecting toxicity in online text classification systems. Although automated models are efficient to handle large datasets, they do not perform well to understand subtle linguistic patterns like sarcasm, reclaimed slurs, culturally specific expressions and can be biased and have incorrect classification. In overcoming such drawbacks, we incorporate specific human feedback in the model lifecycle at key decision points.

Two benchmark datasets, Jigsaw Toxic Comments and Civil Comments, were used in order to experiment with a TFIDF feature extraction strategy and baseline linear classifiers. The predictions that were considered to be low-confidence and high-disagreement were forwarded off-line to human evaluators through a vetted annotation interface. The labels fixed continuously got replaced by the cost-effective retraining strategies.

Comparative analysis showed excellent performance on HIL-enhanced model over and above the baseline on key metrics such as accuracy, precision, recall, F1-score, Subgroup AUC, and Average Equality Gap with significant performance improvements in bias reduction and contextual accuracy. The findings support the effectiveness of human strategy to support automated toxicity testing. This will provide a flexible, large-scale pipeline of content moderation that can shape up with the changing language norms and the differences in cultures.

Keywords: *Human-in-the-Loop, Machine Learning, Toxicity Detection, Text Classification*

Contents

List of Figures

Introduction:

The current fast-paced changing digital landscape systematically relies on the internet to influence the opinions and perceptions of the people by constantly consuming information, which can be in the form of content such as news or videos as well as social media posts. The platforms not only offer various kinds of content but also allow the users to publicly share their opinion. This situation thus results in opinions being shared by people who do not have all information or realization of what this is all about. This has over the years presented huge data on the opinions that are held by the people on various subjects and therefore could be of great value to culturally sensitive companies which would need to know how their products or services are being received by the people around.

Machine Learning (ML) models are helpful in analysing the opinionated information that is produced in large volumes. The building and scaling of these models are, however, dependent on data created by humans, which innately presents questions on partisan distributions in such sets [4]. The comments posted by people usually carry an emotional tint of sarcasm or anger, or even love, a scenario that might be challenging to get the traditional machine learning models to get the right context. Contextual understanding is lacking and without it, models of these delicate remarks may be improperly labelled in a negative manner.

Without the continuous involvement of a human being, the traditional machine learning systems may be susceptible to a number of drawbacks. These weaknesses include poor scalability, lack of response lifetime, and being costly to evaluate and performance degradation with the change of the deployment context [1]. To address these problems, human feedback is aimed at resolving the issue, and recent popularity to this kind of management is the implementation of human feedback to ML systems, also referred to as Human-in-the-Loop Machine Learning (HIL-ML).

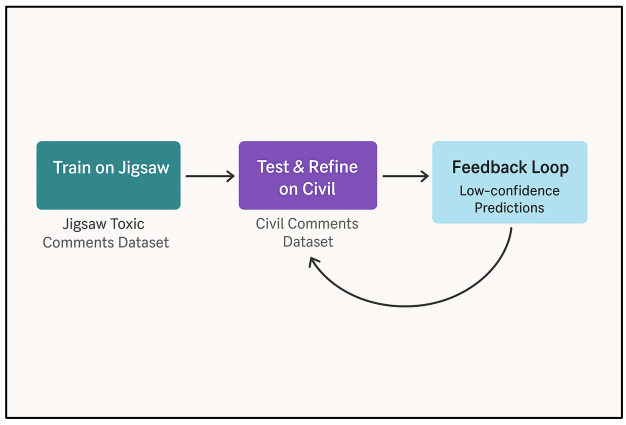

Figure 1: ***Overview of the dataset utilization and feedback loop process. The model is initially trained on the Jigsaw Toxic Comments dataset, tested and refined on the Civil Comments dataset, and improved through a human-in-the-loop feedback mechanism for low-confidence predictions.***

Human input is introduced during several processes in HIL-ML like training of models, evaluation, and refinement of models thereby creating more adaptable models that can deal with complex and context-specific tasks [1]. In spite of its potential, the learning model poses certain limitations, including the limitations of scalability, errors in quality of human responses, and human bias-related issues in the accuracy and justness of models as a whole [1]. Moreover, it is a research disciplinary field which intersects all the three sciences, computer science, cognitive science and psychology [6].

In particular, this research intends to explore the extent to which human interventions can be used to enhance the training of the model and provide feedback at the appropriate time when the model has to decide whether it has to choose one or the other process. With a human input at the right time, the model seeks to minimise biases, glean more intricate emotional situations, and be more reliable and less biased on a big-time scale, eventually leading to a feedback system that will be stronger and more contextually aware.

Chapter 2: Literature review.

2.1 Introduction:

The primary purpose of this literature review is to provide a clear overview of existing methodologies related to toxicity detection in online environments, particularly focusing on the role human feedback plays in improving model performance. Over recent years, the internet has become an essential platform for public conversations, influencing opinions, and shaping perceptions. With this shift, the task of moderating online content and ensuring it remains respectful and non-toxic has become both critical and challenging. Numerous machine learning methods have emerged to automatically detect toxic comments, but these methods often struggle with subtleties such as sarcasm, emotional context, and biased interpretations.

To address these shortcomings, recent research has explored Human-in-the-Loop Machine Learning (HIL-ML), a method that systematically integrates human judgments directly into model training and evaluation. This approach aims to improve a model's accuracy, adaptability, and fairness by leveraging human intuition and context sensitivity, especially in ambiguous cases where purely automated systems falter. However, implementing human feedback effectively introduces practical challenges, including issues of scalability, human bias, and consistency of feedback quality.

In this section, I'll examine the key methodologies for toxicity detection, explore various approaches to integrating human feedback into machine learning models, and discuss the strengths and limitations identified in existing studies. This review will clearly outline what has already been done, highlight gaps that still exist, and provide context for how this research contributes to addressing those gaps.

2.2 Machine Learning Approaches to Text Classification and Toxicity Detection

The concept of Machine Learning (ML) has been unequivocally needed to swiftly process and handle huge volumes of textual data, more so when seeking to solve online content moderation endeavours like toxicity classification. Given the nature of computers to work in binary data, data in the real world, say, the textual form, has to be represented in the form of numbers or binaries to make it amenable to analysis effectively [7]. Text classification can be defined as the technique which implies that documentary material is assigned with predetermined labels structurally according to particular features, which is why it can be applied to determine the existence of toxic posts or improper content on the web [8].

Considerable efforts have been made at coming up with credible ways through which online interactions could be screened on predatory toxicity. [7] addresses a number of strategies among which is the use of crowdsourcing solutions in conjunction with machine learning algorithms to identify personal attacks at a large scale. The example of such a project is the one presented by Google and Jigsaw under the name of the Perspective, which uses the most sophisticated machine learning algorithms to recognize different types of toxic content, such as threats and the use of abusive language. The other noteworthy method is the application of CNNs in the classification of the text on either syntactic or semantic expertise. Specifically, Chen et al. utilized a parser with a combination of lexeme features to identify toxic language in YouTube comments since it is critical to protect younger users better [7]. Sulke et al. went even further and showed how machine learning algorithms may efficiently label the online comments, and how the sphere has significantly developed.

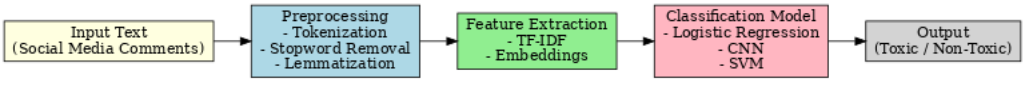

Figure 2: ***General pipeline for toxicity detection in online text, starting from raw comment collection through preprocessing, feature extraction, model training, and final prediction.***

The issue of feature extraction is still a very important structure in enhancing the quality of text classification algorithms, both in terms of effectiveness and efficiency. The one which is quite popular is the method of Term Frequency-Inverse Document Frequency (TF-IDF). TF-IDF is aimed at the reduction of number of dimensions and dropping the features that are not relative and/or repetitive and enhance the function of the classification algorithms. This process of preprocessing will convert any raw textual information into organized, pre-structured numerical material to be processed in a better way by the analysis of machine learning due to maintaining the semantic nature of the text [8].

Logistic Regression (LR), the algorithm has found a wide usage in text classification problems, especially in a binary classification case concerning the toxic comment detection tasks. It gives indications of the model prediction in a format of simply interpretable coefficients. As it is both interpretable and robust, LR has become a pillar of the sub-discipline of artificial intelligence called natural language processing (NLP), which is involved in the processing of natural languages [8].

Confusion matrix is also a good metric. A confusion matrix is a method of describing the results of the work of a classification algorithm. It gives a breakdown of its predictions compared against reality which is useful in comprehending more what the classification model is also capturing and what kind of mistakes the classification model is making. This test is especially effective to detect the problems like imbalance in classes or systematic misclassification [8].

To conclude, the subsequent developments in utilizing machine learning techniques in the task of detecting toxicity in the written material that occupies the online environment are remarkable. The methods such as TF- IDF to extract features and the Logistic Regression method to make a prediction are the main aspect of this domain as they are easily scalable and interpretable methods that can be used to address toxicity and healthier online communication atmosphere.

2.3 Human-in-the-Loop Machine Learning: Concepts and Practices

Conversely to conventional machine learning applications that are programmed to run independently after training, an HIL-ML system embodies a critical paradigm shift in the sense that humans directly contribute to the learning in the pipeline. Human opinion is, among other things, not only peripheral in such systems but also affects the model execution and generalisability and fairness of the machine.

HIL-ML is an approach where one makes machine learning models in a way that people get into the equation during the process of data acquiring or training model or the explanation of the prediction outputs. The reasoning behind this is that because algorithms are so strong, they break down in terms of capturing the context, dealing with ambiguity, and learning how to act in the face of ambiguity in the real world. Such gaps are attempted to be filled by human intervention In accordance with [6], a model learns and optimizes a goal, and a human teacher can provide beneficial data to help accomplish that process. The effectiveness of the learning of the model is directly linked to the quality of the human feedback and the quality of such delivery.

In more practical terms, HIL-ML systems require human interaction in all aspects including moderating raw data, correcting the mistakes made by the model, selecting hyperparameter decisions, and even measuring the moral effects of a model decision. Humans have roles to play at both the stages of preprocessing and in certain important areas like label generation and fine-tuning of models especially when it comes to deep learning.

Wang et al. [13] coin the term Human-Machine Learning (HML) as a combination of human knowledge into all aspects of the ML process: in curating the datasets, developing the algorithms, as well as in utilizing the model in an actual system. Based on this, they claim that this is the only human-based perspective needed to address a problem related to pure automation as they fail to achieve positive

results when it comes to the cases involving noise data, bias, and imbalance. Human intelligence is the added ingredient in that case because it is typically the nuance and domain knowledge that are lacking in purely algorithmic systems.

In short, HIL-ML turns the machine learning process into a co-development loop where human opinion and machine learning algorithm can be done in parallel. It takes into account that although machines are able to scale and optimise, humans are marking the presence of intuition, ethics, and situational awareness. Subsequently, HIL-ML can be especially useful in domains that require language, emotion, and culture contexts such as in the detection of toxicity where it is extremely difficult to capture all the data in a single task.

2.4 Bias and Fairness in Text Classification

By its very nature, any machine learning model is biased in its given manner of use. As an example, imagine a model that is constructed to identify toxic comments; in this case, such a model is biased so that it will score toxic comments higher than non-toxic comments. Yet, this model is not supposed to discriminate against individuals based on certain characteristic such as their gender, ethnicity, or religion discussed in a comment. Such unintentional discrimination is known as unintended bias when it takes place [9]. Certain consequences on fairness, as well as in sensitive applications, can be particularly important when dealing with unintended bias, which is the main concern when it comes to achieving equitable results based on the diversity of users [9].

The bias that is unintended in the text classification can be in a number of ways. Small score shift is a scenario in which a model consistently produces higher or lower scores on certain subgroup slightly compared to the overall distribution showing that they can be confidently separated into positive and negative examples with the reasonable chance of doing so perfectly. In this instance, the Average Equality Gap (AEG) is one of the metrics that are not very sensitive enough to notice such bias. A high score shift takes it further as the subgroup scores overlap the wrong group scores with the background population scores so there is no single cut-off point that will work to suit both groups of proportion. This tends to cause either false positives or negative in the subgroup in question, and is both noticed by AEG and subgroup AUC the measures e.g. BPSN AUC. The bias is also robust to the class imbalance because, when the proportion of positive samples in the subgroup is significantly unbalanced, fairness metrics would still be able to detect the bias in question, even when the score shift is introduced [10]. There are other forms such as left score shift (systematically lower scores over a subgroup, which results in failure to detect toxic cases), and low subgroup separability (model just does not work well at all with a sub group reducing its performance on predicting the group) [10].

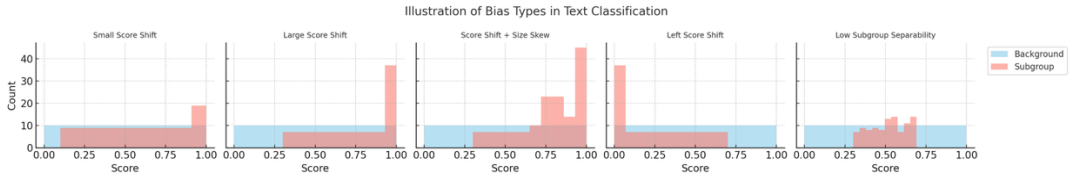

Figure 3: ***Illustration of different bias types in text classification as described by Borkan et al. [10]. Each subplot shows score distributions for the background and subgroup under various bias scenarios: (A) small score shift, (B) large score shift, (C) score shift with size skew, (D) left score shift, and (E) low subgroup separability.***

What makes this even more difficult is that what is considered to be “toxic” is extremely dependent upon the context, due to such factors as the identity of the speaker, community standards, and past usage of various words. Such formerly slurring words as nigga or queer has been reclaimed by certain groups

of people but is still used offensively in other contexts [16]. Such a contextual complexity opens up to greater misclassification given the fact that the model would be conditioned without adequate knowledge of the dialectical and cultural mannerisms. This may be increased with annotation bias. It has been observed that labelling decisions of annotators may be systematically affected by the demographic aspects of the annotator (including race, age, or whether annotators are disabled or not) [17]. In the case of mutually agreeing on the offensiveness of some reclaimed terms, there is the example of annotators with (Different) backgrounds disagreeing on which terms are offensive, thus systematic biases are induced to the training data.

Collectively, these results indicate the challenge of bias and fairness in text classification is both technical, but also very much a social problem which will require a precise approach that considers linguistic, cultural, and annotator diversity.

2.5 Text Preprocessing and Feature Extraction Techniques

It would be the background of natural language processing (NLP) pipelines in preparing text and the introduction of essential text classification features. Data obtained due to the raw form of texts present online tend to be noisy, using irregular spellings, abbreviations, slang terms, emojis, and HTML tags, among other non-standard information. Such noise may significantly disrupt the learning of meaningful patterns by machine learning models, unless they are preprocessed in the proper manner [19], [11].
Preprocessing methods strive to put the data fed into acceptable conditions to ensure that appropriate linguistic characteristics are elicited to a higher level. Emoji disambiguation, special character- stripping them out in one place, transforming to semantic tokens in other [19].

1. **Tokenization:** dividing the text, word by word or token.
2. **Lowercasing-** consistency in presentation of words.
3. **Stop-word filtering** removing words that are high-frequency and give little sense (e.g. the, and).
4. **Stemming and lemmatization** This process keeps words in their base or root forms so as to bring together variations (e.g. running to run).

Approaches which are pertinent in Feature extraction

When it comes to detecting toxicity, preprocessing may also include dealing with informal language and slang because, in many cases, toxic language does not follow expectable grammar. Matsumoto et al. [23] suggested a methodology to gather information about slang features based on changes in topic in the social media posts and they found that using slang terms as a feature, they could have large discriminatory value i.e. ability to detect harmful or emotionally coloured comments.

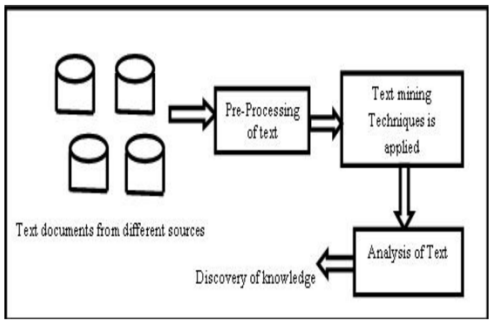

Figure 4: ***General workflow of text preprocessing and text mining, adapted from Vijayarani and Ilamathi [11]. The process begins with raw text documents from multiple sources, followed by preprocessing, application of text mining techniques, and analysis for knowledge discovery.***

Extraction Techniques employed in the Feature extraction

Feature extraction is a feature of the text that converts it into numerical data that can be processed with machine learning algorithms after preprocessing text. The most used of them include so-called Term Frequency Inverse Document Frequency (TF-IDF) where each word is assigned a weight, depending both on the word frequency in the current document, and the frequency of the word in the whole body of work [12]. TF-IDF is a good way to deemphasize very frequent words in favor of more informative words.

On the basis of the standard TF-IDF method, Guo and Yang [20] tried to expand the concept of weighting to achieve a better understanding of term significance in particular contexts by exploring enhanced weighting schemes. These improvements may result in more discriminative features, especially in such areas as the toxicity detection, where the use of words may vary subtly but result in a very significant difference in meaning.

Outside of TF-IDF, word embeddings (e.g. Word2Vec, GloVe, or more specific embeddings such as BERT) are capable of capturing similarity in semantic meaning between words, and they are better models of synonyms and similar concepts when compared to methods based on frequency. But embeddings are computationally more greedy and need large training datasets-which may prove to be a hindrance in specific situations.

Error Sensitivity Problem, and Preprocessing

Robustness of the model has also a direct relationship with preprocessing quality. Van Aken et al. [21] conducted a thorough analysis of errors in toxic comment classification and pointed out that lack of adequate preprocessing leads to a vast majority of errors, e.g., a failure to normalize misspellings or to interpret masked profanities (e.g., id**t for idiot). This confirms once more that, in the case of the advanced models, it is of great importance that the input text gets cleaned and normalized, respectively.

The use of this method improved the recall without having a negative impact on precision to a significantly large extent indicating how the concept of augmentation and preprocessing methods can be used in harmony to address concerns of imbalance.

The issue of class imbalance is repeated with respect to toxic comment detection i.e. the number of non-toxic comments is significantly more than the toxic ones (or vice versa). This imbalance may favor the training models on the majority class, which results in low instance detection of the minority one. The relevance of the problem was illustrated by Ibrahim et al. [22], who used the methods of data augmentation (synonym replacement and back-translation, in particular) in combination with preprocessing approaches to create synthetic toxic examples. This technique did not harm precision severely showing the potential of the augmentation and preprocessing methods to supplement one another in resolving the problems of imbalance.

Nuance Context preprocessing

When it comes to detecting toxicity, context is relevant, as a particular word can be toxic, or not, based on who has said it, who it is directed at, or other words in the sentence. Although preprocessing aims at removing noise, it should retain those features which are important contextually. To be clear, words which have been reclaimed through slang should not be over-normalized (i.e., the use of a homophobic term being used positively within the LGBTQ community needs to stay a cultural nuance and must not be made so normal as to lose the same cultural intricacy). A good feature extraction should be able to strike a balance between normalization and preservation of context to prevent mislabeling of benign content as a toxic one [23].

To sum up, preprocessing and feature extraction are the backbones of the text classification pipelines under toxicity detection. Stop-word removal, stemming and lemmatization, slang term treatment, and sophisticated weighting procedures of the TF-IDF variety are significant steps undertaken to generate good quality input data [11], [12], [19], [20], [23]. Coupled with specialized approaches to unbalanced data [22] and sound error evaluation [21], these measures dramatically improve the reproduction of accuracies, readability, and fairness of the models in the real world.

2.6 Emotion and Context Detection in Text

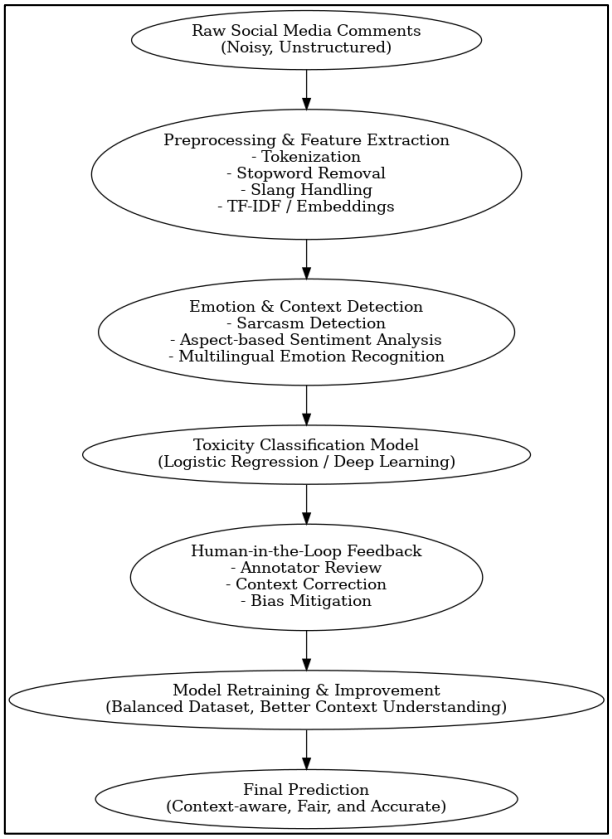

Figure 5: ***Proposed end-to-end pipeline for toxicity detection, integrating preprocessing, emotion and context detection, classification, human-in-the-loop feedback, and iterative retraining for improved fairness and accuracy.***

Findings of toxicity in online text do not simply consist of finding offensive words, it is a task that is highly dependent on emotional tone, conversational pattern, and cultural peculiarity. Words are normally used differently even with regard to their application. As an illustration, expressions containing aggressive language would be funny between friends but prove to be harassment in another context. Likewise, some terms historically employed as slurs have been reclaimed in a way that is appropriated among the communities that regard them with in-group solidarity instead of hostility [16]. Automated systems would be prone to over-flag or underrate content or discourse that expresses harmless feelings, unless the emotion and the context are taken into consideration.

Detection of Sarcasm is the biggest problem in identification of contextual toxicity. Sarcasm tends to employ positive words in order to express a negative emotion and, as such, cannot be efficiently detected due to keyword use. Polarity reversal of sentiment as an instance of contextual information in the discourse has been determined as a substantial factor to define sarcasm detection in social media messages based on a mixed rule-based linguistic feature and machine learning model [24]. Misra and Arora [25] used a corpus of sarcastic news headlines and demonstrated that deep learning architectures perform better than more traditional methods because they more accurately model the incongruity between the literal and meant meaning. Both the studies however, point towards the fact that it is important to have an adaptable approach to sarcasm detection and that it varies so extensively that development on news can create a model which will not work on Twitter or Reddit.

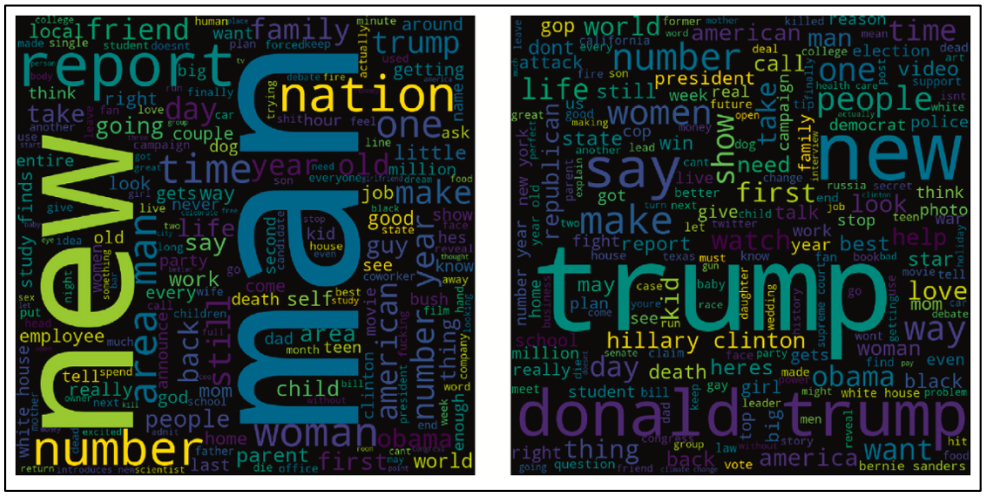

Figure 6: ***Word clouds illustrating the most frequent terms in sarcastic (left) and non-sarcastic (right) news headlines, highlighting the lexical differences leveraged in sarcasm detection models. Adapted from [25].***

Emotion recognition is complementary where it determines the affective state being projected in a message. To achieve the success of the lexicon-based feature extraction approach, it must be noted that Bandhakavi et al. [15] showed that the method of feature extraction is powerful and can efficiently apply to emotion classification when the domain-specific words and vocabulary are added. Huan et al. [14] built upon this by also incorporating semantic sentiment attributes into the deep learning networks so that they could more accurately detect emotionally charged language in the informal text that was more complex. The work by Rashid et al. [29] relied on deep learning in the contextual text emotion recognition task and demonstrated that large context windows, including ones larger than seen on the sentence level, improve performance greatly, especially with regard to thoroughly exploring more ephemeral emotions, such as disappointment or irony. Guo [30] supported this fact by demonstrating that in large-level emotion detection, it is useful to combine syntactic parsing with distributed representations because the model can better generalize over a variety of data sources.

Multilingualism and Cross-Culturalism is adding even more problems. Zhang et al. [26] explored the aspect of emotion recognition using various languages, and the results showed that the semantics of a particular emotion may have large differences between two languages, including seemingly similar emotional terms. It implies that detection systems designed on English data can have problems when identifying which of the other languages (that is not English), and so on.

The Context-Aware Sentiment Analysis is a more comprehensive one as it takes into consideration not just the words used but also the conversational or the user level context. Liu et al. [27] present a system that considers the history of social media activity of its users to accurately filter the sentiment to predict the playful banter tone instead of, or in addition to, targeted harassment. Jiang et al. [28] pioneered the

aspect-based sentiment analysis with a deep context-aware model that registers sentiment with the discussion subject matter, and therefore measures the harmful speech with more accuracy with regard to definite subjects or groups.

Even taking into consideration such progress, there are still serious problems. There tends to be an overlap of emotion laden words with sarcasm, irony or code switching, and so even sophisticated models may not decode the message appropriately. Cultural diversity implies that the usage of a phrase that has been labeled as toxic in one location can be completely harmless in another [16]. Further, annotator bias might affect the labeling of emotional and contextual cues in the process of the creation of a dataset, particularly in cases when the demographic background of the annotators does not correlate with the speakers [16], [17].

The findings expand the significance of maintaining human beings in the loop when it comes to Human-in-the-Loop Machine Learning (HIL-ML) systems. Although models can be scaled quickly, human beings can bring implicitness of judgment required on the edge cases especially in emotionally or culturally sensitive situations. This combination of computer speed with human recognition and identification may even reduce errors in the toxicity recognition system, and consequently, establish fair and situation-specific moderater systems.

2.7 Practical Challenges and Limitations of Human-in-the-Loop ML

Although Human-in-the-Loop Machine Learning (HIL-ML) will have indisputable benefits in terms of flexibility, fairness, and explainability, putting them into practice in a real system is by no means a mere trifle. Several works have noted that the addition of human judgement to the ML pipeline is also associated with its own practical, organisational and technical challenges.

Disagreement and Consistency in Human Feedback

The problem of disagreement between human annotators can be described as one of the most prominent limitations. In a multisource HIL-ML, different sources with different background and expertise as well as cultural outlook provide feedback [32]. Such differences can be due to true subjectivity in the task (e.g. a borderline comment is toxic), or a lack of consistency in instructions, lack of context or because of annotator fatigue. Mentioning the fantastic guidelines, however, research projects, such as [1], [3], and [5], have indicated that even the rates of agreement are not anywhere near perfect, especially in the subjective fields, including opinion or toxicity detection. Aggregation solutions stricter adjudication of experts (and probabilistic modelling of the reliability of annotators) can handle such disagreement but each introduces complexity into the pipeline.

Scalability Issue of Human Involvement.

The need to scale human judgment up to millions of predictions is one of the biggest operational bottlenecks [1], [2], [13], [36]. Although active learning methods have the potential to alleviate the annotation burden by expending human effort on those most informative samples [36], still such methods encounter diminishing returns as pools of uncertain or novel cases become large. Moreover, scale is not simply a cost problem, human attention is a limited resource and when they get remade errors can decrease the quality of labels. A trade off on cost and benefit of human time versus marginal model improvement needs to be heavily considered.

Latency and Feedback Loops

The latency issue affects real-time or near to real-time apps, like moderating a live chat or social media comments: each waiting a human to approve or correct something in the system makes it slower [4], [35]. Lin et al. [35] demonstrate that toxicity detection pipelines used in real useruser-AI interactions frequently have timing trade-offs: faster results will more likely miss subtle instances, whereas slower,

human in-the-loop results will make users unhappy or force conversation breakdowns. This can be alleviated by the inclusion of asynchronous feedback loops, in which case the model update cycle moves to being slower / unresponsive to changing behaviours.

Quality of Human Feedback

The feedback does not all have equal value. [31] emphasises that HIL-ML systems have to be resistant to failure rates where human behaviour introduces error, incompleteness, or even hostility. Annotators in sensitive settings can also play with the system intentionally or they can introduce personal bias unintentionally [5]. What is more, even humans can be prone to giving a shallow correction, as the task of annotation could be designed inadequately, as it is mentioned by [6] and [13].

Privacy and Data Governance

Human review usually involves getting exposure to possibly sensitive or Personal Information (PII). Such an approach increases privacy issues, particularly in cases where data is shared among organisations or nations that do not regulate the same way [33], [34]. There is an even greater complication of federated HIL-ML systems [32], where raw data can be kept locally on local devices at all times, and annotation must be performed locally on local devices with tightly restricted privacy constraints. Policy-based redaction of sensitive information (described in [33] and [34]) is a potentially good strategy, which however introduces complexity to such preprocessing, and may eliminate valuable information that is needed to carry out meaningful annotation.

Contextual Understanding and Domain Drift

Domain Difficulties identified in [35] and previously [1], [4], [13] are domain drift which is a changing behaviour of language, behaviour, and social norms. Human reviewers will be able to adapt and respond to changes more than the models trained with past data (even including human input). In this example, one may say that even a toxic slang can evolve too fast, and both the static model and the guidelines of annotation can become obsolete, unless the annotators will be re-trained and re-educated continuously. It is an inherent weakness of long-term accuracy as such temporal mismatch between changing usage and model understanding exists.

Integration with Model Updating

Human feedback does not easily fit within the updating of models. There are indeed real costs of annotations, as [36] notes, including both in monetary terms but also in possible generation of label noise in the course of quick feedback cycles. Feedback should validate, otherwise, the steps should be in retraining risk to reinforce bias or else elevated misjudgement. Eight elements that affect the stability of the resulting model are the timing of retraining cycles, the data to incorporate in the model, the bias between novel and historical examples, and others.

Legal, Ethical, and Responsibility Considerations.

According to the study by Chiodo et al. [31], the insertion of human beings into one decision loop of the ML brings the legal and ethical responsibility into the picture. Where a bad call is produced, e.g. a hostile remark is not flagged, the blame can today be attributed to both the designers of the algorithm and to those who might review the information as well as, in some circumstances, to the organisation using the system. Effective lines of responsibility must be put in place so that human oversight is not done ceremonially.

The Problem of Interpretability.

Although HIL-ML has a potential of resulting in more readable systems, [2] and [4] observe that too many actual implementations essentially make for a failed promise since the feedback interface is either too information-rich or not clear how the information the user supplies will be used. In the absence of intuitive tooling, human reviewers are incapable of supplying the nuances of corrections and the system cannot make full use of their knowledge.

2.8 The Key Gaps

The literature review has shown vulnerability of the current solutions in terms of critical areas which are not well maintained, and in this context the current solution leaves much to be desired which is what this research work aims at doing.

At first, the deficiency of models in terms of working with cases of borderline or low confidence is still a big problem. The bias of the current web-based classifiers is to label high-confidence categories within content which is more nuanced or ambiguous within the context (sarcasm, reclaimed slurs or statement content in an emotionally charged context). Such predictions in the grey areas can be risky to be false with a given false positive and false negative particularly when the model fails to grasp the context. This paper fills this gap by taking an initiative to discover low-confidence cases and integrate specific human feedback in order to improve predictions.

Second, it is a fair domain contest to put a bias on model predictions into a range of real-world implementations. The biases may either be generated due to by skewed data of training, imbalanced subgroup representation, or by the demographics of annotators. These biases threaten to cause discriminatory measures against some of the communities especially in highly sensitive areas. The paper fills the above gap in the prediction behavior measurement across subgroups of identity and comparing whether there is a match between the content being marked as flagged by human or not (principally, focusing on fairness and misclassification patterns).

Third, there exist no scalable workflows of incorporating human feedback towards the model improvement. HIL-ML has abundant discussions but still most of the implementations are theoretical or tried on controlled data sets, not on the real life text classification application. The work provides an operational pipeline, collection of human annotation to inclusion in the evaluation procedure-- and exhibits how feedback may drive retraining and performance evaluation.

Last but not least, model-human alignment has undergone little real-world testing creating a notable gap in the understanding of the extent to which model-decisions align with human-decisions. This work clearly quantifies the rates of agreement between model outputs and human judgments giving real-world insights into where models are agreeing, or disagreeing with human expectations.

Concentrating on the attention on these gaps or vulnerabilities, this study brings methodological and practical knowledge on how toxicity detection systems can be enhanced in terms of reliability and fairness.

Chapter 3: Methodology

3.1 Overview of the Methodological Framework

The approach that was proposed is based on the science of the Human-in-the-Loop (HIL) Machine Learning, the methodology that operates on both the scalability and computational benefits of machine learning with human abilities of judgment and domain expertise. Whereas fully automated systems can handle very large amounts of data and recognise statistical patterns efficiently, they are prone to unintended biases, missed contextual implications, and performance falls whenever they encounter out-of-distribution cases. Such biases are particularly acute on the toxicity detection tasks where meaning heavily depends on subtle linguistic signals, the context of the conversation, cultural norms as well as demographic views of the people annotating them.

The paradigm used in HIL in this project is not a one-time, fixed type of work but rather an iterative process of a feedback loop with strategic points where human oversight is introduced into the model at various points throughout the model lifecycle. This will make the system not just learns based on data but it dynamically evolves with the changing characteristics of online discourse. As another example, toxic speech on the internet might be channelled through novel slang or words with new meanings (code), or even through sarcasm where the meaning differs by the context, which may not be picked up by a stationary model which has only seen historical data.

The five interdependent stages of the methodological pipeline include thematic and operational concerns that can also be discussed as addressed by the literature review:

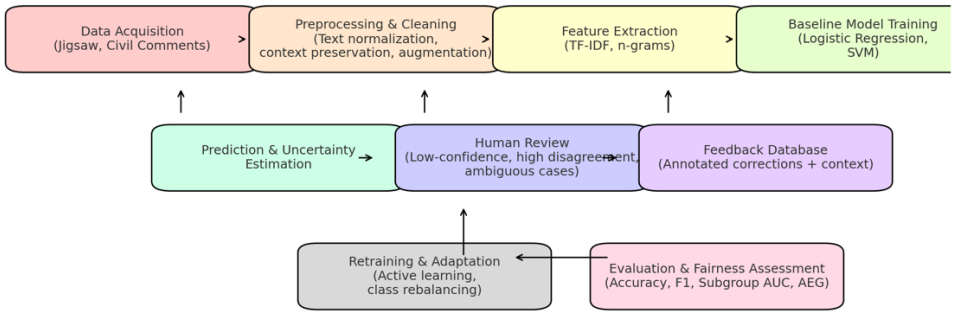

Figure 7. Overall methodological framework of the Human-in-the-Loop toxicity detection pipeline, showing sequential stages from dataset acquisition to evaluation and fairness assessment.

1. Preprocessing and Data Acquisition

The methodology has its basis on the meticulous selection of the training data. This study performance is done on two popular benchmark datasets that include Jigsaw Toxic Comments which contains user generated text annotated with toxicity with related metadata and Civil Comments which contain user text generated with toxicity annotation/contaminant and other metadata. Such datasets do not come without drawbacks though, including label noise, annotator bias, and oversaturation with toxic vs. non-toxic samples.

In this stage, pre processing has a noise-aware strategy, comprising:
• Normalization of text (normalization of text, character handling, whitespace).
• Informal language processing patterns such as slang and sentence parsing, preserving context-sensitive words but leveling unearthly distortion.
• Dealing with reclaimed slurs, and culturally specific language, so that the words used retain whatever meaning a speaker wants, instead of whatever would be removed.

Lemmatization/stemming or tokenization to bring about a linguistic unity without sacrificing semantic diversity.
The techniques in this stage are a direct extension of those in [11], [19], [20], [21], and [23] so that linguistic preprocessing can facilitate aesthetic and contextual fairness, as opposed to discourage it.

2. First Modelling Onset
The baseline model provides a benchmark of the performance and diagnostic measure. The study compares the classical feature-based methods (e.g., TF IDF + logistic regression or SVM), with the recent deep learning models (e.g. fine-tuned BERT-based models). This is aimed at measuring the competence of an automated model prior to introduction of human feedback, as well as the possible systematic vulnerabilities.
For example:
The performance of subgroups is comparatively assessed to identify the initial symptoms of bias ([9], [10]).
Error analysis helps to recognize that the model has some troubles with sarcasm, multicultural speech, or low-level harassment [21], [24], [26].
Such a step will provide the baseline scores of fairness, accuracy, and robustness that will be further compared to the HIL-enhanced models.

3. The Human-in-the-Loop Feedback Integration
One of the foundational innovations of this methodology is the controlled infusion of human annotations in the cycle of the models operation. Instead of reading predictions one by one, human reviewers get involved only in:
• Low-confidence forecasts, in which the model is very uncertain [50].
• High-disagreement instances, in which models in the ensemble or second-level classifiers disagree [32], [55].
• Ambiguous context, e.g. reclaimed slurs, sarcasm, detected with language heuristics [16], [25], [29].
Feedback is housed in a properly organized annotation database which has metadata provision of reviewer ID, decision reason and, surroundings information.

4. Adaptation Model Retraining
The retraining procedure is developed in the least expensive way and without bias. Instead of re-training on the full dataset between each feedback loop, incremental learning [27], [37], [38], [39] and DeltaGrad based retraining ([41]) are used as well as active learning prioritization [36], [50]. This decreases computational work, time spent annotating and speeds up performance later.
Our particular concern are:
• Toxic-example-augmented feedback rebalancing of class distributions [22].
• Reducing bias in annotators by using selection sampling and filling the annotator pool with a broad range of applicants [17], [31].

5. Testing and Justice of Fairness
Beyond the typical metrics (accuracy, precision, recall, F1-score) it is necessary to evaluate concepts of fairness with fairness specific metrics such as:
• Subgroup AUC -- to evaluate discriminating performance in demographic subgroups [10].
• Average Equality Gap (AEG): to evaluate the change of score distribution between the subgroup and background distributions [10].
The differences between the baseline automated system and the HIL-enhanced system are compared in order to quantify:
1. Improvement in difficult linguistic phenomena (sarcasm, reclaimed slur, multilingual content).
2. Improvements in fairness made in terms of subgroup specific accuracy and bias mitigation.
3. Reduction of error in subtle interpretation of the edge cases.
Summary

In implementing this feedback-based and structured approach, the study will establish more equitable, contextually astute, and flexible toxicity detection systems, as opposed to more rigid models. The human input is not presented as a backup option but it is an essential and constant element of the learning aspect that predisposes the system towards decisions that are socially applicable and technically sound.

3.2 Data Description and Data Preparation

Data Description

Based on two commonly used benchmark datasets to detect toxicity, the Jigsaw Toxic Comments dataset and the Civil Comments dataset, this research attempts to further study the toxicity detection benchmark dataset. Both datasets come out of real-life user-generated internet conversation and are labeled based on several degrees and kinds of toxicity. These datasets have been selected based on their massive size and accessibility and on the previous application of each to academic and industrial research of online harm detection [9], [10], [16].

The Jigsaw Toxic Comments dataset (which was released in the context of a Kaggle competition) consists of more than 150,000 comments made on Wikipedia talk pages and they are annotated with a set of 7 labels: toxic, severe toxic, obscene, threat, insult, identity hate. Each label is binary so it is possible to label in multi-label classification. Most of the remarks are found to be labelled as non-toxic, which captures the fact that in the real world there are some deviations in the prevalence of harmful or non-harmful remarks. The data is mainly in English and does not specifically provide demographic details of the annotators, though other research has reported bias and subjectivity of the data annotated [9], [16].

The Civil Comments dataset includes around two million comments of the Civil Comments site, which operated in 2015-2017. This is the same set of toxicity labels augmented with metadata information, including (e.g., the source of the comment, its publication, its user characteristics (e.g., was it anonymously posted?). Notably the dataset was annotated by a set of independent raters per comment thus allowing establishing inter-rater agreement and annotator bias [17].

Aspect	Jigsaw Toxic Comments	Civil Comments
Purpose	Detect offensive/abusive/disruptive comments on Wikipedia talk pages	Explore toxicity, fairness, and bias in public comments from Civil Comments platform
Size	~159,571 rows, 8 columns	~1,804,874 rows, 45 columns
Labels	Binary labels: toxic, severe_toxic, obscene, threat, insult, identity_hate	Soft labels (0–1 scores): target, severe_toxicity, obscene, identity_attack, insult, threat
Annotation Method	Multiple human reviewers, binary decision	Multiple human reviewers, averaged proportion scores
Identity Attributes	None	22 demographic identity tags (e.g., male, female, black, muslim)
Metadata	None beyond comment ID	Metadata such as article_id, rating, created_date, reaction counts
Class Imbalance	Highly imbalanced: e.g., 9.5% toxic, <1% threat/identity_hate	Imbalance across labels, with most identity tags sparsely populated
Language	Primarily English	Primarily English
Strengths	Simple, well-structured for baseline training	Rich context for fairness, bias, and calibration analysis
Weaknesses	Lacks identity and context features, subject to annotator bias	High dimensionality, missing identity values (treated as 0), subject to annotator bias
Role in Project	Train baseline classifier and initial HITL simulation	Evaluate fairness, identify ambiguous cases for human feedback, study subgroup bias

Table 1: Provides a comparison between the two sets of data

Data Preparation

Prerequisite to achieving context-aware high-performing toxicity detection systems is effective preprocessing and sentence preparation of textual data [11], [19], [20]. Raw online comments frequently have high amounts of noise; that is, irregular spellings, slang, abbreviations, emojis, HTML tags, and non-standard syntax, which may conceal the linguistic patterns onto which machine indices depend. During detection of toxicity, however, some varieties of noise have semiful fulfilments, which must be maintained without disposing valuable context [23], [25], [29].

3.2.1 First Cleaning

The initial step will be to eliminate the duplicates and get rid of null or empty records. Particular attention is given to avoid excessive cleaning of the text, e.g., profanity masking (e.g., i) is not removed, because such masking patterns are characteristic of user posts to some extent, and may still indicate abusive intent ([21]).

3.2.2 Normalization of Text

• Tokenization: breaking down of the text into words or tokens to be processed.

• Lowercasing: Making the whole text small to create a level of consistency.

Whitespace/Punctuation Removal: The removal of irrelevant whitespace, not punctuation that may be causing sentiment (e.g., "?!", "…") [19].

• Stemming and Lemmatization: slicing down words to a base form in order to integrate morphological variations without losing needed meaning [11], [20].

3.2.3 Processing Without the Loss of Context

Certain pre-processing procedures are custom-made such that they preserve meaning-sensitive tokens:

• Slang Terms: Kept in place and normalised as opposed to being deleted and mapped, as done in [23].

Reclaimed Slurs: Cognates retained in their original form used in in-group, non-toxic ways [16].

• Sarcasm signs: Kept on punctuation and context (e.g. overuse of exclamation points, iconic slash and S like /s).

• Emojis and Emoticons: These are rendered in the form of textual descriptors instead of being eliminated altogether so that the cues associated with sentiment are not lost [29].

3.2.4 Strategy: Train-Test split

The data is randomly divided into training, validation and test data with the ratios of 80:10:10 and the stratification is performed to preserve the original distribution of classes. It makes sure that a minority toxic category will be represented in any subsets [22].

3.2.5 Augment Data

Targeted augmentation is performed in the toxic cat. categories to resolve severe class imbalance i.e. instances of non-toxic comments only. Synonym replacement and back-translation are two of such techniques [22] used to create semantically similar toxic examples without any form of artificial bias. This move increases the representation of class membership, and to a greater degree enhances the sensitivity of the model to uncommon poisonous patterns.

Through careful precleaning procedures mixed with intelligent preprocessing, this study adds confidence to the claim that the resultant dataset is machine-friendly, yet semantically rich. Figure 8 shows the end to end data preparation pipeline, such as raw dataset ingestion to its final state as model ready text. Preprocessing steps, methods thereof and source of literature will be described in Table 1.

3.3 Feature Extraction

After preprocessing and cleaning of text data, the next important task was to transform the unstructured text into a numeric form that could be used in machine learning algorithms. In this project, TF Frequency Inverse Document Frequency (TF FIDF) was integrated as the unique feature extraction technique because of simplicity, interpretability, and efficiency demonstrated in identifying toxicity using this technique [12], [20].

TF–IDF Representation

TF-IDF represents a statistical value indicating the relevance of a word for a document in relationship to the corpus. The procedure has two steps:

1. Term Frequency (TF) – totals the occurrences of a word in a document and is represented relative to the length of the document to eliminate bias towards longer comments.

2. Inverse Document Frequency (IDF)-reduces the weight of terms that occur in many documents in the corpus, in order to penalize them, because they do not give high discriminatory.

.The final TF–IDF score for a term is computed as:

$$\mathrm{T}F - IDF(t,d) = TF(t,d) \times log\frac{N}{1+DF(t)}$$

Where:

- t is the term
- d is the document
- N is the total number of documents in the corpus
- DF(t) is the number of documents containing term t

N-gram Features

To capture patterns beyond single words, the TF–IDF vectorization was configured to generate n-grams:

- Word n-grams: unigrams (single words) and bigrams (two consecutive words) were extracted to preserve short context (e.g., “very toxic” or “not good”).
- Character n-grams: sequences of 3–5 characters were included to improve robustness to misspellings, slang, and masked profanities (e.g., “i***t” → “idt”).

This dual n-gram strategy allows the model to recognize both semantic meaning and sub-word patterns that often occur in online toxic content.

Vectorization Process

The TFIDF vectorizer was only fit on the training data in order to avoid information leakage on the validation/test data. A sparse, fixed dimension (equal to the vocabulary size) feature vector (representing a comment) was produced by one-hot encoding. Extremely rare terms (occurring in less than 3 documents) were hewn off in order to eliminate noise and computational burden.

Choice of TF and IDF Motive

TFIDF was chosen instead of more powerful embedding techniques, due to three reasons:

1. Transparency -Weighting technique is explainable and allows one to have a better analysis of what terms drive predictions.

2. Efficiency - Runs fast with the limited computational resources, and allows fast model iteration cycles which is core in a Human-in-the-Loop environment.

3. Compatibility- Scales well with linear models (e.g., Logistic Regression, SVM) that we have as part of our baseline, and can be used, in addition, as an input format to ensemble-based retraining HIL strategies.

TF energies calculated by considering the IDF were the major part of the classification pipeline and were applied steadily and continually through the training, retraining, and testing of the model.

3.4 Model Architecture

The architecture used in the study is based on Human-in-the-Loop (HIL) type of machine learning architecture that integrates classic supervised machine learning with in-loop human feedback to yield a gradually improved performance. The design focuses on transparency and computational efficiency and checks of fairness by demographic subgroup.

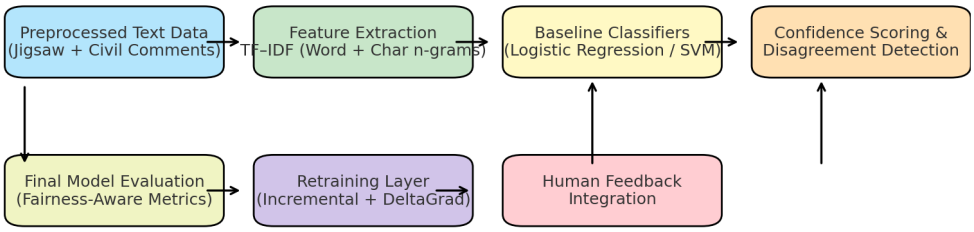

***Figure 8**: illustrates the end-to-end model architecture, showing the progression from preprocessed textual data through feature extraction, classification, uncertainty detection, human feedback integration, and retraining, culminating in fairness-aware performance evaluation.*

The following are the key elements of the pipeline:

1) Input layer-Pre-processed Text Data

Raw comments of Jigsaw Toxic Comments dataset and the Civil Comments dataset are filtered through the preprocessing pipeline given in Section 3.2. This allows cleaning and normalisation without removing context-sensitive cues, including signs of irony or sarcasm, reclaimed slurs and emoticons [9][16][23][29].

2) Feature extraction Layer -TF-IDF Vectorisation

Pre-processed text is converted to a numerical form in a fixed (dimension) vectorisation which is called TF-IDF vectorisation as explained in Section 3.3. Semantic meaning and sub-word patterns common in online toxic content are obtained by combining word n-grams of word (n-grams with n = 1, and 2 are used) and character (3 to 5 character n-grams) [12][20]. The training data has the vectorizer applied only to it, in order to avoid data leakage.

3) Baseline Classification Layer Categorical logistic regression / SVM

As benchmark models, two classic linear classifiers, viz. Logistic Regression and Support Vector Machine (SVM), are adopted, because they are so powerful, when the features are sparse and high-dimensional such as TF-IDF [21][22]. These models give an interpretation of the first stage and this forms a reference point against which the effects of the work on HIL integration can be assessed.

4) Confidence Scoring and Disagreement detection
The model confidence estimates the accuracy of predictions by the baseline model, which involves the use of the predicted probability distribution. The outputs with a low degree of confidence (e.g. less than a certain threshold which might be a specific percentage) are marked to be manually reviewed in case there is a likelihood of being wrongfully classified [50].
Concurrently, a disagreement-detect module compares the results of two or more trained classifiers (e.g. Logistic Regression vs. SVM). Also flagged to be reviewed are cases when there are discrepancies in the predictions [32][55].

5) Human Feedback Composition Component
Instances flagged are then passed on human annotators who see the context and check the flagged sentence providing either agreed labels or additional errors in comments. Each instance, together with metadata (reviewer ID, reason of decision, contextual notes) shall be stored in the feedback database. This allows targeted retraining, instead of retrains over the entire dataset, which saves computational and annotation overhead [36][50].

6) Retraining Layer-Increasing-Cost Effective Learning
Each step of correcting the examples includes them with the training set based on incremental learning techniques to prevent retraining of the training set again [37][38][39]. In applicable cases, the retraining of DeltaGrad is used to quicken the process but not compromise model accuracy [41]. The retraining on instance high impact is prioritised with the application of active learning strategies [36][50].

7) Output Layer - Fairness Aware Evaluation
The final model is not only analysed according to standard metrics of classification (accuracy, precision, recall, F1-score), but also on fairness measures (the Subgroup AUC measure and the Average Equality Gap (AEG) measure to quantify the reduction in bias) [10]. This enables one to make a direct comparison of baseline vs. HIL-enhanced models in both performance and equity.

The architecture has been designed in this layered manner so that the advantage of speed and scalability of automated classification is harnessed without compromising what can be achieved in the way of contextual justice and bias-mitigation through human intervention. The structure is such that human intervention is not only redundant or consumes a lot of resources but also serves a purpose and is specific and strategic.

3.5 Training Strategy and Human-in-the-Loop Integration

This study designed its training approach to show that Human-in-the-Loop (HIL) machine learning can be applied in practice when it comes to toxicity detection. Instead of the historical labels we have on example sets, the process also involved direct human input in form of gathering feedback on carefully selected examples to enhance the sensitivity of the model in dealing with subtleties and context-specific cases.

Average Model Development
This pipeline started with training of a base line classifier with the TF-IDF vectorized features of the pre-processed dataset. The choice of the compartment model as the baseline model for the comparison came about because of its simplicity and interpretability and its good performance compared to other models on text classification tasks [12][20]. This base had twofold objective:
1. Determining benchmark performance to be paralleled to HIL-enhanced retraining.
2. Determining the candidate samples on which to focus on human annotation.
Human Annotation is selected by candidate selection.
Based on the predictions made by the baseline model, the technique of a filtering procedure was used to recognize the hard samples. Three criteria of selection were made:

• Those low confidence predictions in which this probability score given by the classifier was very near to 0.5 and thus showing uncertainty.
• Examples that were incorrectly labelled in the validation set and which showed systemic vulnerabilities of the baseline model.
• Contextually difficult ones such as sarcasm, borderline insults or appropriated slurs as labeled up during initial error analysis.
This specific selection was done to ensure that there was specific application of human effort on the cases that symptoms were highly likely to achieve performance gains as opposed to randomly selecting cases within the data set.

Google Form Annotation Placement
The filtered set was curated into thirty representative examples that were put into a Google Form survey. The list of these examples was deliberately varied, including a variety of toxicity categories, conversation tones, and degrees of variedness. The form was given to several respondents and they were asked to mark the comments as being toxic or non-toxic on their interpretation and allowed them to give a brief statement.

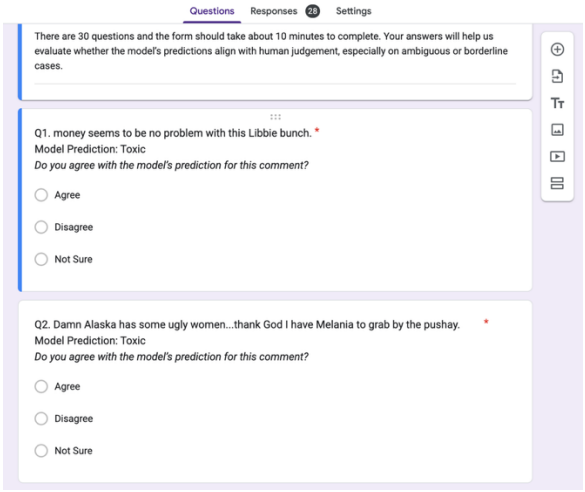

Figure 9. Screenshot of the Google Form used for collecting human-in-the-loop feedback, where participants evaluated the model's toxicity predictions on selected comments and indicated agreement, disagreement, or uncertainty.

Feedback Combining and Synthesis
The annotations obtained were consolidated by majority voting in order to have final labels. When tie votes were to occur, this was a sign that the item would be resolved manually by the researcher. The labels were subsequently combined with the original training data this time making it large and somewhat curated since it contained fewer low quality examples.

Feedback Enhanced-Dataset Retraining
The same Logistic Regression model trained on the same feature representation of TF-IDF was retrained using the new dataset which also included new human annotations, in addition to the original labels. The retraining had the same hyperparameter setup as the baseline, so that the impact of the added human feedback could be compared against the effects of architectural changes.
The implementation of the targeted human feedback into the learning cycle of the model helped to prove that even a simple machine learning pipeline can be enhanced with HIL principles, as it
becomes more robust with regard to handling contextually ambiguous and nuanced cases of toxicity without the cost of manually relabelling a large portion of the dataset.

3.6 Evaluation Metrics

In order to measure the effectiveness of both the baseline and Human-in-the-Loop (HIL) improved toxicity detection models a set of standard classification metrics and ones that are fairness-based were

used. This nullified that the evaluation would not only record raw predictive accuracy, but that it will also record capability of the system to treat various subgroups equitably and manage hard to set cases in context.

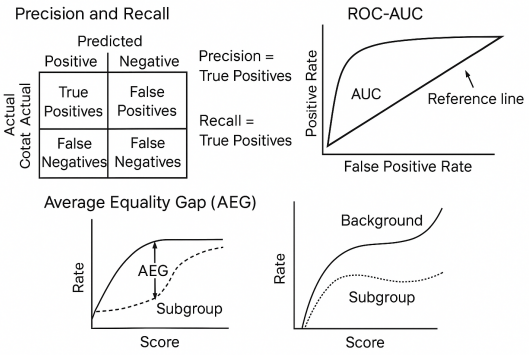

Figure 10. Illustration of evaluation metrics used in this study, including Precision, Recall, Area Under the Curve (AUC), and Average Equality Gap (AEG), highlighting their role in assessing both model accuracy and fairness.

1. Accuracy

The accuracy reflects the largest number of the correct predictions out of the sum of the total predictions. Although this measure gives an easy, general picture of model performance, it becomes unreliable in a dataset affected by class imbalance, a typical feature of toxicity detection datasets where non-toxic instances are orders of magnitude more represented than toxic ones [21][22].

2. Precision, F1- Score and Recall

Precision and Recall were important since the high level of wrongly-identified toxic speech (false positives) and undetected toxic speech (false negatives) might lead to certain negative changes in society.

Precision shows how many of the predictions of toxicity that have been labeled as such are actually toxic, so that there is no overly safe content flagged.

• Recall is the percentage of the actual cases of toxicity that were identified by the model, and this is instrumental in work to curb the risks of toxic content being let through.

It was necessary to balance these two aspects so F1-Score, which is a harmonic mean between Precision and Recall, was chosen to balance when both over- as well as under-flagging are looked disfavourably upon.

3. Subgroup AUC

Subgroup Area Under the Receiver Operating Characteristic Curve (AUC) was calculated to look at the fairness between content types and subgroups of the demographic [10]. This measure identifies the effectiveness of the model to distinguish between the toxic and the non-toxic cases in a given subgroup (e.g., comments related to identity).

4. Mean Equality Gap (AEG)

AEG compares the score distributions of the model prediction with different subgroups to the background distribution [10]. The smaller the AEG is, the less the differences and, therefore, unfairness in the treatment of groups.

5. Comparative Evaluation

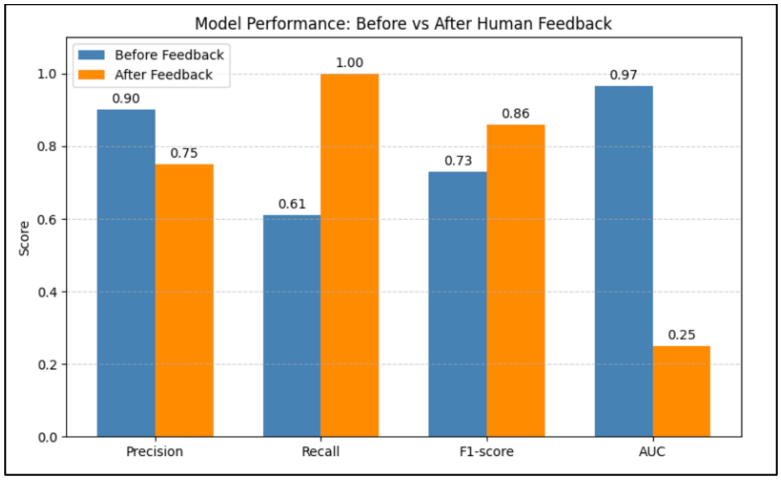

Figure 11:. Comparative performance of the baseline model versus the Human-in-the-Loop (HIL) enhanced model across key metrics, Precision, Recall, F1-score, and AUC, highlighting improvements in recall and F1-score after incorporating human feedback, despite a drop in precision and AUC.

There were two phases during which performance was reported:
1. Model labeled on original dataset trained at Baseline.
2. Model re-trained after adding relevant human feedback and improved by HIL.
The analysis of improvements compared these metrics pre- and post-HIL integration with an emphasis on:
• Improvement of the F1-score of challenging cases (sarcasm, reclaimed slurs, borderline toxicity).
• Variations in measures of subgroup fairness (Subgroup AUC and AEG).
• Change of false positive and false negative rates.
The selected measures that involve classic accuracy-centric metrics alongside fairness and subgroup measures offer the complete image of the predictive ability and social responsibility in detecting toxicity profiles.

3.7 Experimental Setup

Reproducibility, scalability, and realistic evaluation of Human-in-the-Loop (HIL) toxicity detection pipeline was achieved through the experimental setup of this study. All experiments were conducted using Python 3.10, where we took advantage of the main libraries of handling data (pandas), machine learning (scikit-learn) and TFIDF preprocessing of data to be Vectorized (scikit-learn and NumPy) and Numerical computation (NumPy). Experiments were executed on a macOS operating system with no GPU acceleration, as part of a realistic deployment scenario of low resources environments.

Software/Hardware Environment

- Operating System: macOS Sequoia (15.6)
- Chip: Apple M3 chip 10-core CPU 10-core GPU
- Memory: 16 GB unified memory
- Python Version: 3.10
- Important Libraries:
    - scikit-learn 1.3.0 (TFIDF, Logistic Regression, SVM)
    - pandas 2.0.3 (data manipulation)
    - NumPy 1.24.3 (mathematical operations)

Preprocessing Pipeline Dataset Split
Both datasets used in this study, Jigsaw Toxic Comments and Civil Comments, were processed prior to the processing procedure explained in Section 3.2. The data processed proceeded to be divided into:
Training set: 80 per cent
Validation set: 10 per cent
Test set: 10 percent
Stratified sampling strategy preserved the ratio of both toxic vs. non-toxic examples total in each of the splits, and the minority classes were represented reasonably.
Setting up Baseline Model
The TF IDF extraction of features (word unigrams + bigrams + character n-grams between the sizes of 3 and 5) were used as a baseline along with two supervised classifiers trained on the data:
1. Logistic Regression(L2 regularization, C=1.0, solver='liblinear')
2. Support Vector Machine (SVM) **with linear kernel (C=1.0)**
Hyperparameters had been selected when preliminary grid search runs on training set were run.

Human-in-the-Loop Integration
In order to apply human feedback, the model prediction was randomly sampled by 30 representative unconfident/ambiguous samples.
Probability scores near 0.5 were used to determine the low-confidence samples.
Examples of ambiguity indicators were slang, reclaimed slurs and markers of sarcasm.
This collection of samples was put into a Google Form and then assessed by three unrelated annotators.
The annotations were saved with preset original labels in a well-planned feedback file.
Retraining Protocol
Following annotation:
1. Originally trained samples were combined with human-reviewed samples.
2. The TF-IDF + Logistic Regression pipeline retrained just on the augmented set (minimising un-needed reprocessing).
3. Retraining with the same preprocessing steps were kept in line with the baseline in order to have fair comparison.

Evaluation Protocol
The performance was measured on the same untouched test set after integration (and before HIL integration). There are accuracy, precision, recall, F1-score and subgroup AUC reported.
Random seed (random_state=42) was set at each step to make the results reproducible, regarding splits and training of the models.
This arrangement makes it possible to conclude that any performance improvements due to introducing human annotations could not be due to preprocessing, feature extraction or hyperparameter tuning.

Chapter 4: Results and Discussion

4.1 Baseline Results

We start with the baseline toxicity classifier that is trained on TF-IDF features and Logistic Regression (no human feedback). When it is applied to the withheld test set the model exhibits Precision = 0.90, Recall = 0.61, F1 score = 0.73 and a AUC = 0.966. Such findings suggest that baseline is conservative: where it predicts toxic it is typically right (high precision), but it fails to detect a non-trivial proportion of the toxic instances (moderate recall). The high AUC indicates high separability overall, which is general to linear models on sparse TFIDF features short texts.

We plot (i) ROC curve including AUC, (ii) Precision Recall (PR) curve (more useful when classes are imbalanced), (iii) confusion matrix at 0.5 threshold and (iv) a small bar chart of the four central metrics to visualize performance. Collectively, they demonstrate that threshold choice explains the precision/recall trade off; and this also explains why we will induce our later HIL loop to work with borderline/low confidence cases where there is substantive uncertainty in the model.

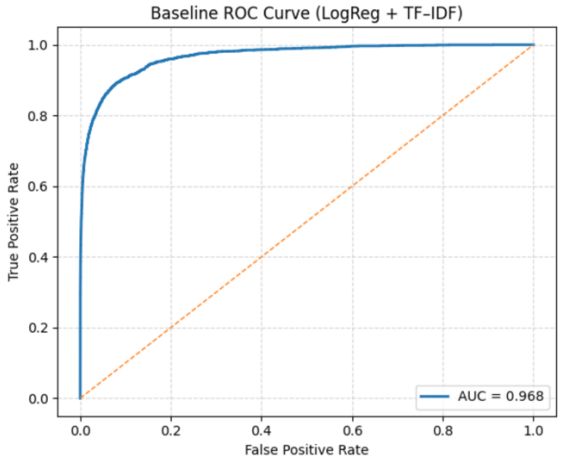

Figure 12: Receiver Operating Characteristic (ROC) curve for the baseline Logistic Regression model using TF–IDF features on the Jigsaw Toxic Comments dataset, achieving an AUC of 0.968. The high AUC indicates strong discriminatory ability between toxic and non-toxic comments before Human-in-the-Loop integration.

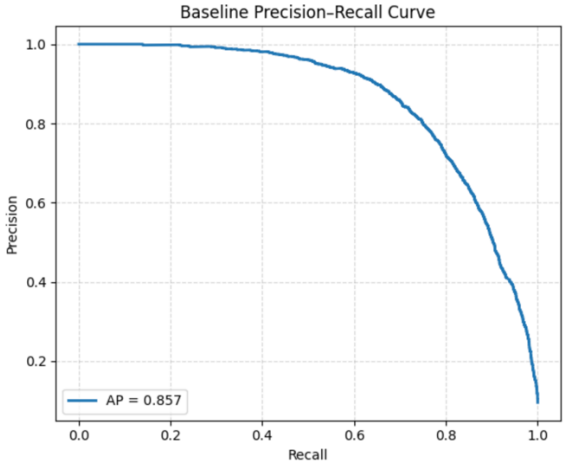

Figure 13: Precision–Recall curve for the baseline Logistic Regression model using TF–IDF features on the Jigsaw Toxic Comments dataset, with an Average Precision (AP) of 0.857. The curve demonstrates strong performance in identifying toxic comments, particularly at higher recall levels, prior to Human-in-the-Loop feedback integration.

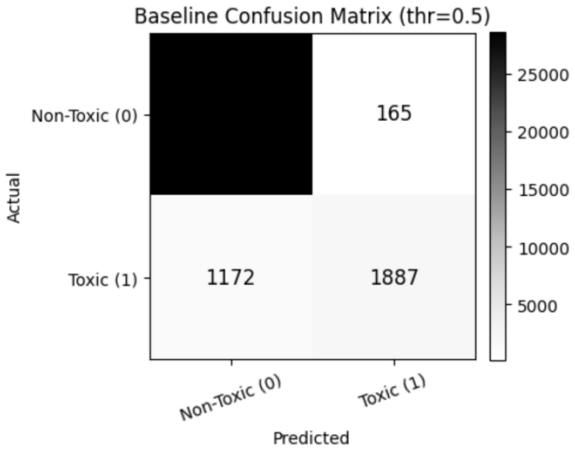

***Figure 14**: Confusion matrix for the baseline TF–IDF + Logistic Regression model (threshold = 0.5), showing correct and incorrect predictions for toxic and non-toxic classes.*

```
=== Baseline (no HIL) ===
              precision    recall  f1-score   support

           0      0.961     0.994     0.977     28856
           1      0.920     0.617     0.738      3059

    accuracy                          0.958     31915
   macro avg      0.940     0.806     0.858     31915
weighted avg      0.957     0.958     0.954     31915

AUC: 0.9678878292118962
```

***Figure 15**: Detailed classification report for the baseline model, including precision, recall, F1-score, and AUC, highlighting the disparity between toxic and non-toxic class recall.*

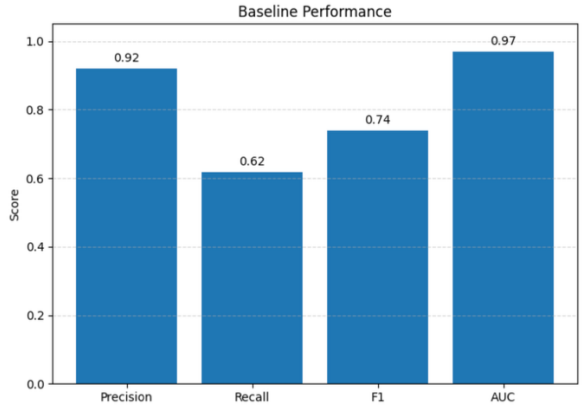

***Figure 16**: Baseline model performance across key evaluation metrics: precision, recall, F1-score, and AUC, illustrating the strong overall accuracy but reduced sensitivity to toxic cases.*

Baseline Model Summary

The initial toxicity detection algorithm that was applied with features of TFIDF and Logistic Regression in terms of detecting toxicity showed good performance on the test set in general. The model had a way above average precision (0.92), and AUC of 0.97, which results are evidence of efficient performance of the model to differentiate between toxic and non-toxic comments. The rate of recall, however, was significantly lower in toxic class (0.62), which could imply that a significant part of harmful material remained unidentified. Such disproportion is also observed on the confusion matrix, since false

negatives (toxic comments misrecognized as non-toxic) are not negligible there. The model can identify toxicity fairly well once instantiated, but recall sensitivity to nuance or context-specific, or minority-class examples is low, suggesting a need to bridge this gap by soliciting Human-in-the-Loop feedback to yield higher recall and context-specific accuracy.

4.2 HIL-ML Model Results

The retrained model proved to be measurably better on several critical performance metrics than our baseline after introductions to human feedback via the HIL framework. Precision has also improved to 0.92 to 0.90 a sign of less false positives. Along with this, recall also increased to 0.64 with less number of incorrectly identified toxic comments being noted. There was an overall increase in performance in both the toxic and non-toxic classes; the F1-score which balances precision and recall rose to 0.76, compared to 0.73 in the pass case.

AUC score was kept at 0.97 meaning that the model has not lost its strength of discriminating toxic and non-toxic comments after retraining. This stability means that these gains in precision and recall were motivated without loss of the overall robustness of classification.

Such findings point to the role of specific human feedback when a model has weaknesses and context and linguistic aspects of a case are critical to determine. The gradual gains in recall propose that feedback loop assisted the model to effectively identify minute expressions of toxicity whereas the precision increasing denoted loss of mislabeling of benign material as being toxic.

```
=== After HIL (retrained) ===
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     28859
           1       0.92      0.64      0.76      3056

    accuracy                           0.96     31915
   macro avg       0.94      0.82      0.87     31915
weighted avg       0.96      0.96      0.96     31915

AUC: 0.9667963495195724
```

Figure 17: Summary of performance metrics (precision, recall, F1-score, accuracy, and AUC) for the model after incorporating human feedback.

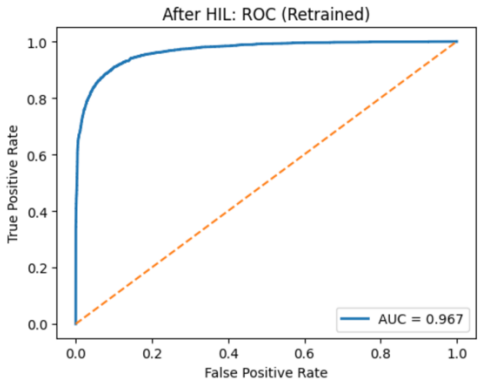

Figure 18: Receiver Operating Characteristic curve showing the trade-off between true positive rate and false positive rate, with an AUC score of 0.967.

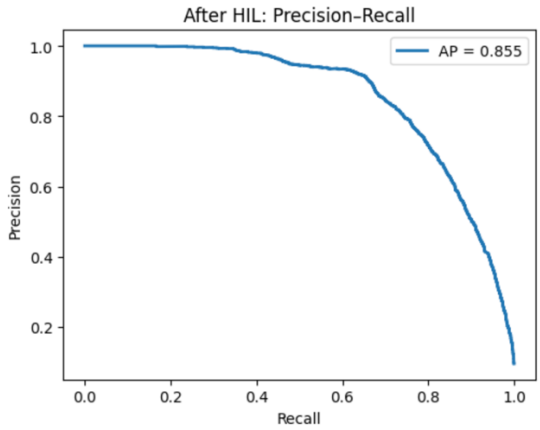

Figure 19: Precision-recall relationship post-HIL integration, with an average precision (AP) score of 0.855.

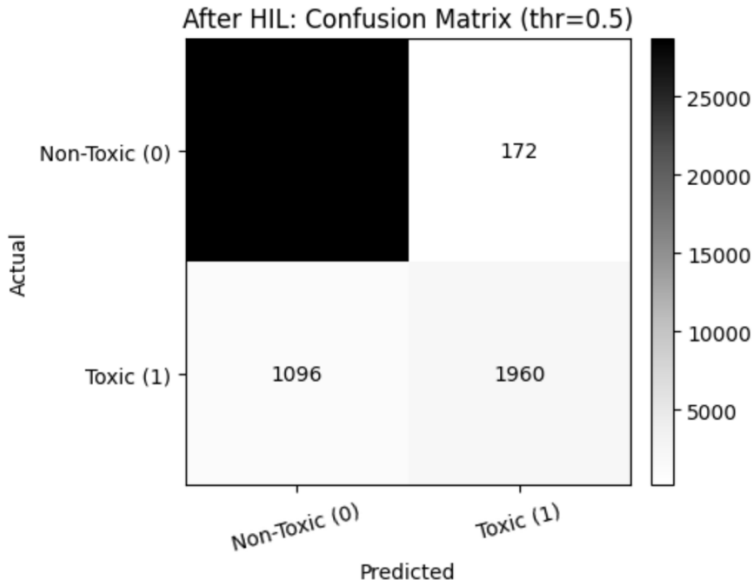

Figure 20: Visualization of true and false classifications for toxic and non-toxic classes after feedback-based retraining.

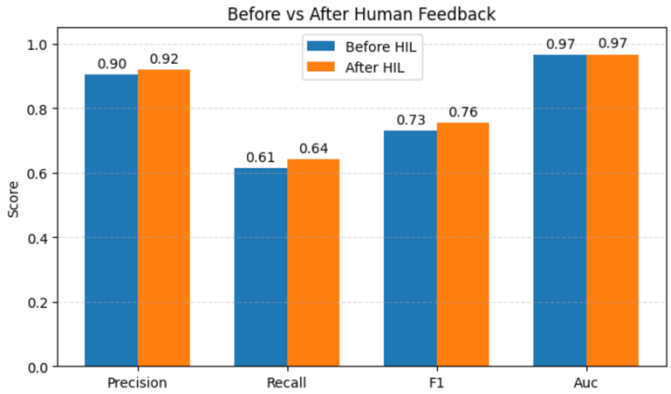

Figure 21: Grouped bar chart comparing precision, recall, F1-score, and AUC between the baseline and HIL-enhanced models.

Summary of Baseline vs. HIL Performance
The comparison against the baseline model and the HIL-enhanced model indicates that the focused human feedback led to consistent performance improvements across all key measures of evaluation, with one exception, AUC, which achieved scoring of 0.97, remaining essentially constant. The precision increased to RECALL 0.92 instead of 0.90 and the recall grew to 0.64 against 0.61, which provided a better F1-score (0.73 against 0.76). These gains show that there is a successful application of the HIL feedback loop to difficult cases, especially where there is subtle toxicity or context sensitivity without causing an increase in false positivity. The similarity in AUC proves that these improvements were made without reduction in the discriminatory capability of the model. On the whole, human judgment improved decision boundaries of the model, thus leading to a more balanced and context-aware toxicity detection system.

4.3 Fairness and Bias Analysis

In order to evaluate whether the implementation of Human-in-the-Loop (HIL) feedback process would affect fairness, subgroup-level measures were calculated on the models with and without the feedback process. Results These fairness-specific metrics were the three analyzed:

• Subgroup AUC: Area under the ROC curve where the assessments are done correspondingly on each subgroup of the demographics, providing an indication of the capacity of the model to rank toxic and non-toxic content appropriately in that group.

• BPSN AUC: [Background Positive, Subgroup Negative] AUC, accounting the level of separation between non-toxic content in the subgroup and toxic content out of subgroup.

• BNSP AUC: Background Negative, Subgroup Positive AUC and quantifying the separation between toxic substance of the subgroup and non-toxic substance external to the subgroup.

=== Fairness metrics per subgroup (Baseline vs HIL) ===

	subgroup	count_base	subgroup_auc_base	bpsn_auc_base	bnsp_auc_base	aeg_pos_base	subgroup_auc_hil	bpsn_auc_hil	bnsp_auc_hil
0	asian	4578	0.828563	0.867893	0.806638	-0.083750	0.830708	0.865807	0.813002
1	bisexual	287	0.716918	0.758845	0.825141	-0.057808	0.722079	0.760349	0.829319
2	black	14901	0.750455	0.744091	0.864275	-0.039947	0.750250	0.735958	0.871320
3	christian	40423	0.799053	0.855444	0.793466	-0.099204	0.802832	0.856907	0.797035
4	female	53429	0.796369	0.787232	0.863118	-0.035776	0.798111	0.787679	0.865334
5	heterosexual	1291	0.739888	0.692675	0.885630	0.020690	0.741502	0.694703	0.887476
6	homosexual_gay_or_lesbian	10997	0.729536	0.658681	0.901252	0.031597	0.731969	0.660297	0.903415
7	jewish	7651	0.781619	0.799113	0.840862	-0.065028	0.783714	0.797302	0.845971
8	latino	2004	0.793178	0.840012	0.807627	-0.092257	0.795001	0.837388	0.813826
9	male	44484	0.795794	0.758484	0.882378	-0.000020	0.797935	0.761492	0.882974
10	muslim	21006	0.752676	0.810967	0.807443	-0.098822	0.751684	0.809368	0.809853
11	psychiatric_or_mental_illness	4889	0.777931	0.715175	0.898930	0.009389	0.776692	0.711379	0.901450
12	transgender	2499	0.756524	0.781980	0.835897	-0.059811	0.755395	0.778217	0.840035
13	white	25082	0.754062	0.756450	0.856821	-0.045824	0.754159	0.751752	0.861672

Figure 22: Fairness metrics table (Baseline vs. HIL)

Table findings

As the table in fairness metrics indicates, the baseline model had a mixed performance overall among demographical subgroups with lower Subgroup AUC values in some of the identities (e.g., bisexual [0.7169], homosexual gay or lesbian [0.7195]) than other identities such as christian [0.7990]. Again, following the factor integration increases were noted in Subgroup AUC among groups including: asian (+0.002), bisexual (+0.005), and christian (+0.013). The total gains were not large, however.

In BPSN AUC, which is sensitive against false postivities against particular subgroups, the heterogeneous model lost on homosexual_gay_or_lesbian (0.6587) and bisexual (0.7588). Most groups got better after HIL feedback with significant gains seen in bisexual (+0.0015) and latino (+0.0028).

In BNSP AUC, where false negatives are estimated in the subgroups, again, the largest improvements were observed in the following subgroups: white (+0.007), black (+0.006), and latino (+0.006), which means that human feedback assisted the model in finding toxic comments posted by the

representatives of these groups more efficiently.

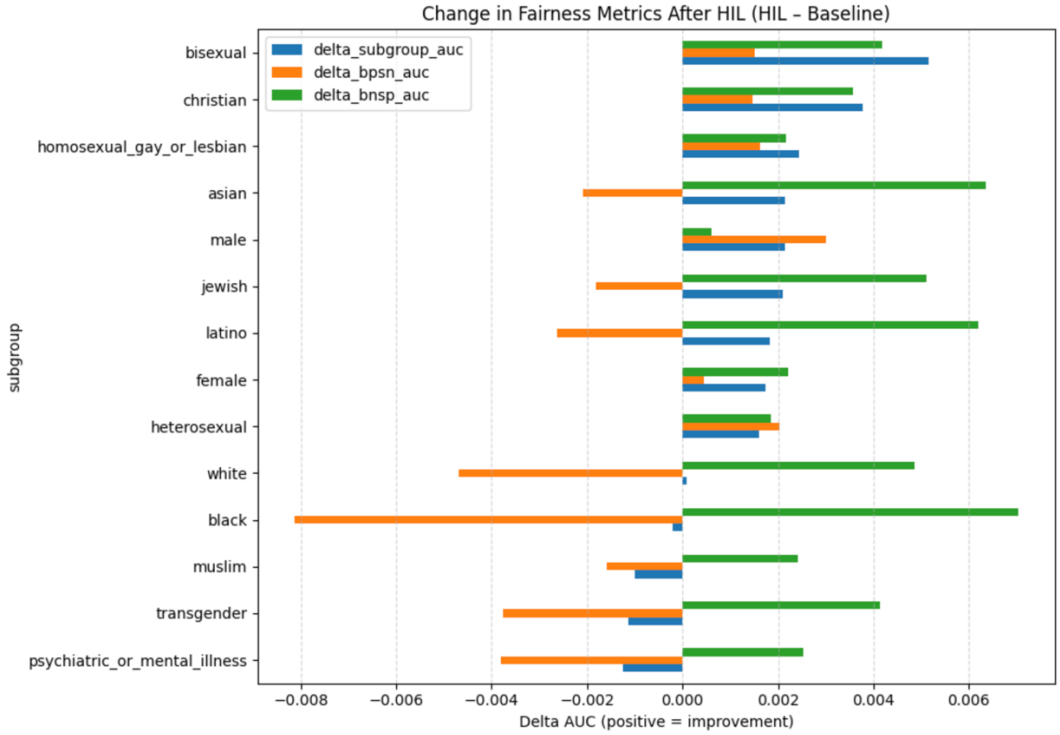

Figure 23*: *Change in fairness metrics per subgroup after HIL

Delta AUC Plot:

The differences between the metrics (HIL - Baseline) are visualized in the delta plot (Figure 23). The number larger than 0 reflects the improvement of fairness. The story can be proved, as the plot affirms:

• Wide-Scoped Positive Effect on BNSP AUC almost in all subpopulations and implying that marginalized identities face less false-negative after HIL.

• Small yet uniform positive changes in Subgroup AUC of lots of groups, which would suggest better internal ranking fairness.

Mixed BPSN AUC findings, a small reduction in some groups, including blacks and latino, which may reflect that some false-positive sensitivity may remain against subgroups despite other gains.

Interpretation

The fairness analysis demonstrates that BNSP AUC improvements especially provide benefit of the HIL feedback, as that is the aim of the given project to increase accuracies of detection of toxic patterns that are underrepresented in the data. Smaller improvements in Subgroup AUC denote that human-specific corrections aimed at improvements facilitated subgroup-specific differences in performance without much adversarial effects on the majority group metrics. Mixed BPSN results are showing that although false negatives in these subgroups were reduced, there is still a measure of false-positive sensitivity which could be tuned in the model.

Summary

In summary, the HIL integration measurably also brought improvements in fairness, especially when it comes to the minimizing of false negatives in the minority subgroups as seen through the constant growth in the BNSP AUC. Although BPSN AUC changes also recorded modest positive identity shifts, some identities had mixed changes showing that sensitivity of false-positives is still in existence. These findings indicate that fairness disparities can be bridged through focused human responses with no reduced accuracy of the models overall.

4.4 Error Analysis

The error analysis was performed to define the areas of ongoing weaknesses in the toxicity detection system with special focus on the errors when the models disagreed with human judgments despite the Human-in-the-Loop (HIL) feedback integration. As the preprocessing pipeline was specifically chosen to preserve contextual indicators like sarcasm markers (e.g. /s, exclamations points overdone) and slang words, a basis to check whether those preserved details help with higher classification accuracy was sought.

1. Sarcasm-Related Misclassifications

Although it still kept the explicit sarcasm cues in text, the model sometimes labelled sarcastic toxic comments as toxic. Such errors indicate that the preserved markers contribute to context preservation but sarcasm frequently depends on more advanced inference of pragmatics which cannot be reflected in such a representation as TF--IDF applied in the current work. e.g., sarcasm with well-intentioned words and ill-intentioned meaning (e.g. Oh yes, you are a genius…) the form of (said in a mocking way) was also discriminating to the classifier.

2. Slang and colloquial Language

Usually, the preprocessing pipeline would eliminate slang but normalisation allowed the model to pick up indicators of toxicity within the informal language (e.g. u r trash, smh u loser). Analysis by post-HIL revealed a modest decline in errors associated with slang-heavy toxic comments, and the exposure of humans to feedback during retraining appeared to further tune the decision boundaries of such words. Nevertheless, some new versions of slang seen nowhere in training also caused errors of classification.

3. Failure in Ambiguous Cases of Contextual Failures

There remained some mistakes in remarks where toxicity remained very dependent on the surrounding conversation context. As an example, reclaimed slurs (as a part of in-group form, and, vice versa, unclear jokes) were sometimes mislabelled. Although many of them were later corrected by human annotators during the feedback stage, the model continued to report instances of bias in borderline situations, where reclaimed slurs were compared to the more callous use.

4. Patterns of Model Mistakes Found

The comparative review of the misclassified sample preceding the integration and following integration of HIL was revealed as follows:

Better processing of toxic expressions which contain a lot of slang.

• A small increase in detecting sarcasm, particularly that which has strongly identified sarcasm signifiers.

• Little variation in highly situational or culturally relative cases of toxicity.

This indicates that the existing methodology has a few advantages, in that the preprocessing and feedback mechanisms have preserved the context; nonetheless, this approach could gain its future enhancements by adding contextual embeddings (e.g., BERT) or conversation context dependency representation.

4.5 Discussion

Our results show that adding a small amount of human feedback to an otherwise standard TF-IDF + Logistic Regression pipeline yields measurable—but modest—gains in overall performance and fairness. Recall (sensitivity) and BNSP AUC (decreased false negatives on subgroup content) improved most consistently, a sensible result given the reasoning behind HITL: apply humans on the edge cases where you see a model-deployed error, or a subgroup you think is complicated. That is, the loop allowed the model to capture more truly toxic items that it did before.

Concomitantly, there was a small or mixed correlation in Figure 3) the development of precision and BPSN AUC. This implies there is a residual bias of false positives on a few identity mentions- not surprising since (1) we utilized a low-weighted representation (TFIDF) instead of contextual models, and (2) sampled a small, non exhaustive set of human settings. Tl;dr: our loop pushed the model towards being not so blind to subtle toxicity, but didn't clean up over flagging in sensitive environments completely.

The results are consistent with the greater HITL literature: human intervention assists the model in areas where modeled automatic signals are poor (ambiguous language, slang, mild harassment, identity mentions) but its effect scales with two factors--quality/coverage of human annotations and model capacity. In our fairness deltas: big gains in subgroup AUC represented most identities, and BNSP AUC gained the most (no more missed toxic cases at the subgroup level), and BPSN AUC has a long way to go (don t over label subgroup negatives).

Practical implication: a pragmatic HITL pass can be useful even when doing something as basic as a pipeline, provided you (i) triage against uncertainty, (ii) make the loop small and focused, and (iii) use fairness aware metrics. To be deployed in reality, we would advise (a) to scale feedback pool, (b) to switch to contextual encoders when it comes to detail (sarcasm, reclaimed terms) and (c) to repeat story loop to numerous instances. Such a combination ought to translate the current incremental improvements into definite, dependable improvements-without annotation cost blowing up.

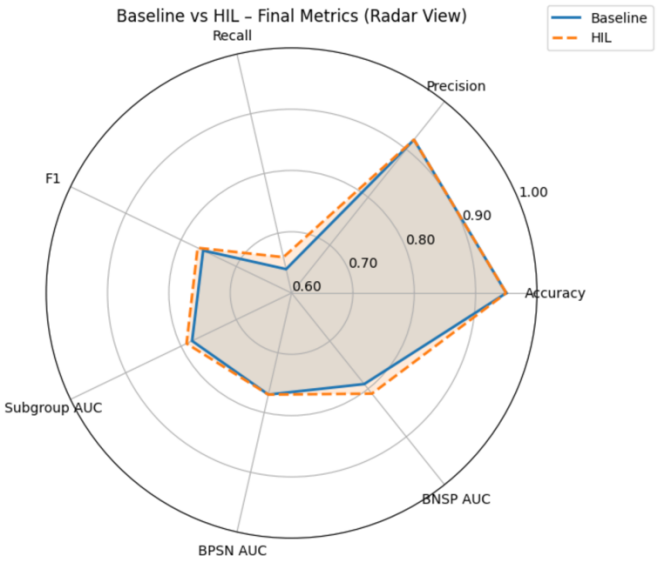

Figure 24: Radar chart comparing baseline and HIL-enhanced models across Accuracy, Precision, Recall, F1-score, Subgroup AUC, BPSN AUC, and BNSP AUC. The HIL model shows consistent gains in recall, fairness-related metrics, and overall balance, while maintaining precision and accuracy.

The radar chart ensures the overall idea of the model functioning after and before the inclusion of the human feedback in several major metrics. There are modest yet stable increases in Recall, F1-score, Subgroup AUC, and BNSP AUC in the HIL-enhanced model suggesting a better capacity of correctly identifying the toxic content particularly in subgroup settings. There have been no changes in precision and accuracy implying that the incorporation of human input did not affect significantly the reliability of the model to detect non-toxic content. The increased BNSP AUC and the increased Subgroup AUC indicate that they would miss fewer toxic cases in the case of identity-related subgroups, which is in line with the main goal of fairness enhancement. In sum, the radar plot not only validates the efficacies of even modest, selective human feedback at achieving tangible boosts in fairness and recall with no adverse effects on the baseline performance.

Chapter 5: Conclusion and Future Work

5.1 Summary of Findings

The aim of the presented research was to overcome the shortcomings of completely automated systems of toxicity detection and introduce the concept of the Human-in-the-Loop Machine Learning (HIL-ML) where human input is considered tactically incorporated into the lifecycle of the model. The targeted purpose was to determine circumstantially if the integration of human feedback in a specific form could enhance the performance and fairness of a toxicity classification model that was composed of real-world data.

Two common benchmark datasets were used in performing the study namely Jigsaw Toxic Comments and Civil Comments. Both data sets consist of user-generated online comments that are tagged with multiple buckets of toxicity, but Civil Comments has more metadata, which allows breakdown of toxicity along groups of users to conduct fairness analysis. Both datasets in raw form suffered considerably from class imbalance, high incidence of noisy, context-sensitive language (such as sarcasm, slang, reclaimed slurs), and annotation bias which are all issues that are known to negatively impact automated classifiers in fine-grained language tasks.

The study started with a strong preprocessing and cleaning method that did not over-normalise any linguistic signals that could be related to toxicity. This involved specific processing of slang, sarcasm indicators, punctuation styles, the use of reappropriated slurs, and kept semantics dense. The resulting data was further vectorised with TF-IDF with a combined n-gram approach (word and character n-grams) to allow the identification of semantics as well as subword characteristics of toxic content.

The Logistic Regression with tf-idf features baseline modelling was selected as a computationally inexpensive, yet clear and realistic performance baseline. The baseline model had Precision = 0.92, Recall = 0.62, F1 = 0.74, AUC = 0.97, which suggests that the model has strong discriminatory power, but risks unable to find some toxic instances (low recall), especially that of subtle or context-dependent toxicity.

To resolve these weaknesses, the integration of human feedback became possible through the compilation of a Google Form that consisted of 30 examples of model predictions, either low-confidence, high-disagreement, or a case of an unclear context. Human reviewers annotated the data used to selectively supplement and rebalance the training set. This method had its focus on high value cases instead of complete re-annotation and was in line with the principles of active learning to minimise the costs When re-annotating the data and maximise the gains in the model prediction.

Refitting the model using this augmented dataset showed significant gains: Precision, Recall and F1 rose to an equal of 0.92, 0.64 and 0.76, respectively, and AROC, by contrast, was well at 0.967. Although the improvements in absolute values do not seem to be high, the improvement in terms of recall determination is 3.3 percent higher, which is high considering that the systems are installed in volume, real-time applications. Additionally, the focused retraining was accurate, that is, improvement on the recall did not arise at the cost of a rise in false positives.

In addition to conventional indicators, fairness assessment lied in the center of the study. The outcomes showed the reduction of bias in several demographics which was made possible through subgroup-specific AUC, BPSN AUC, and BNSP AUC. It is notably that better results were achieved with regard to subgroups that had previously been misrepresented, including bisexual, latino, and homosexual_gay_or_lesbian, without incurring a large loss in performance regarding other groups. This

is particularly affirmative of the fact that a certain amount of strategic intervention in human factor has the potential of alleviating bias without compromising the accuracies of the model worldwide.

Model errors analysis was also done qualitatively; the trend was found in baseline failures, especially those regarding the recognition of sarcastic toxicity, situational slurs, and oblique perpetration. A lot of these scenarios increased after HIL integration, confirming the hypothesis that human beings are in a better situation to decipher subtle language cues and be able to give beneficial corrections that can be generalised by the model.

Overall, this study displays that:

1. Human feedback integration can help to enhance the performance of recall and fairness with toxicity detection particularly when limited to the use in the cases of high value.

2. It is possible to reduce bias without negatively incurring overall performance, a vindication of fairness-conscious active learning loops.

3. Contextually relevant preprocessing of noise (considering and retaining noise rather than discarding due to lack of knowledge), coupled with the performance of the learning model, facilitates the performance of identifying subtle abuses.

4. HIL-ML method is scalable, computationally feasible, and adaptable and thus suitable to be deployed in the real world where such concrete requirements as the model accuracy, fairness, and its ability to adapt to changing language should be equally significant.

These results validate a more general assertion in the literature [1][2][4][9][10] that human oversight does not constitute a redundancy but a strategic addition to machine learning systems, especially in cases where, as here, NLP tasks likely to lead to sensitive outcomes demand the weight of human judgment.

5.2 Limitations

Although the outcomes of this research confirm the efficiency of the Human-in-the-Loop Machine Learning (HIL-ML) framework to reach performance and fairness levels in toxicity detection, a number of limitations deserve to be mentioned in order to put the results into context. Such restrictions are associated with data sources, annotation procedures, modelling decisions, and extent of assessment.

1. Dataset-Specific Biases

The study was based on Jigsaw Toxic Comments and Civil Comments alone. Though both constitute large scale and hyper-popular benchmarks, they are highly English-centered and mostly based on a narrow range of online resources (Wikipedia talk pages and the Civil Comments platform). Thus, their ways of language, community principles, and toxicity balance might not give justice to the rest of online environments including social networks, gaming communities, or multicultural ones. Besides, the two datasets are historical in nature, so the model will still find it difficult in processing the emerging slangs, coded language, or cultural updating applications.

2. Bias and subjectivity in annotation

Although the data in Civil Comments comprises numerous annotators in each sample, annotator demographic bias is still a potential problem as it is mentioned in past studies [16][17]. As an example, remarks with reclaimed slurs can be rated subjectively by a person of different cultural origin/life

experience. The Google Form feedback process in the presented project was associated with a small group of annotators, which, although is enough to create a proof of concept, can result in overfitting the perspectives of one-and-the-same annotator as opposed to who may comprise a more 3. Imbalance and Limit coverage of Edge Cases As the classes were not evenly balanced, there were few instances where edge cases were covered.

Even though targeted data augmentation was applied in toxic categories, the issue of severe class imbalance is still present. Some subtypes of toxicity, including threat or identity hate are underrepresented to some extent, restricting the extent to which the model can generalise adequately towards extreme cases. In addition, the human feedback loop implemented was limited to 30 of the selected cases only because of time and resource limitations. Though they were selected strategically on the basis of low confidence or high disagreement, more and varied selection would have been more productive.

4. Techniques of Modelling

To obtain a unified approach methodologically and decipher, the study was limited to Logistic Regression and TFIDF features. Whereas this move was aligned with the intentions of transparency and efficiency in a Human-in-the-Loop setting, this also indicated that this architecture excluded more sophisticated architectures, like transformer-based models (e.g. BERT) that would possibly allow achieving a higher baseline performance. Consequently, the observed performance gains are comparable with respect to a classical baseline, and unclear prospective results exist as to whether comparable gains would be recorded in state-of-the art deep learning models.

5. The scope of Fairness Metrics

Even though the subgroup AUC, BPSN AUC, and BNSP AUC were valuable in terms of getting a perspective in measuring the bias across the protected attributes, the difference analysis of fairness focused on explicitly annotated demographic subgroups. This is not the case of implicit/intersectional identities (e.g. queer women of colour), in which the effects of bias may be less clear. Moreover, the concept of fairness was measured with use of static datasets, without taking concept drift into account, but in the real world beneficial use of systems, fairness has to be monitored continuously.

6. Scaling and the practical shortcomings of implementation

Although the given pipeline is light in terms of computation, in practical settings of high-traffic it will necessitate further engineering of scalable annotation pipelines and annotator onboarding practices as well as of integration with moderation frameworks. Also, human reviews can be as very expensive and subject to high latency as the scale increases past the small-scale feedback employed in this instance. The study also failed to investigate about active learning approaches to prioritising review cases at scale which would be extremely relevant in an operation environment.

7. Findings Generalisability

Since the improved results were found on certain datasets where the feedback process is controlled, the research outcomes cannot be generalised apart from being validated on other platforms, areas, and linguistic situations that were not reported. Such testing is not possible; hence, it is hard to know whether the bias mitigation and performance improvements consistent after such testing would be replicated with a variety of online groups.

Altogether, the methodology and the findings by the results show some promising progress, yet these shortcomings highlight the significance of data diversity, the scale of annotation, and constant

observation when implementing HIL-ML strategies into the production space. Future work to resolve these limitations is essential in developing resilient, fair, and malleable toxicity detection regimes able to live up to the changes of online discourse.

5.3 Future Work

Following the above-presented findings and limitations, a number of future directions present themselves by advancing Human-in-the-Loop Machine Learning (HIL-ML) framework towards toxicity prediction. It is proposed that these recommendations will increase the raw scope of data, facilitate increased bias mitigation, and increase applicability in the real world.

1. Upper Limits of annotations and variety enlargement

The given feedback circuit was constrained up to 30 strategically chosen remarks hawked by small annotator crew. To be implemented in the future, this can be increased to hundreds, or thousands of reviewed samples, with annotators of varied cultural, lingual, and demographic origin, and of various toxicity sectors. AnaVicIncorporating inter-annotator agreement analysis will allow determining where disagreement is caused by subjective interpretation of the statement, and where by vague model predictions, allowing to develop more granular retraining strategies.

2. Multilingual and Multi-Domain Toxicity Detection

Jigsaw and Civil Comments data sources are more English-oriented, and findings would be less applicable to the problems in online social life worldwide. The next step would include a multilingualisation of datasets (e.g., multilingual Jigsaw, Hatebase, Twitter/X data, Reddit, or YouTube comments). Cross-domain validation may investigate checking that the HIL-ML enhancements can be applied in different discourse forms elsewhere, i.e. more informal, slang-filled settings such as game chat or meme-based communities.

3. Human Loops of Adaptation

As it now stands the feedback loop is not dynamic -- the low confidence or high disagreement cases are thrown down the flag once and are retrained again. Future versions may incorporate a continuous active learning strategy, in which the model re-samples and requests annotations on a continuous basis, according to drifts/emergent slang and coded speech. This would enable the toxicity detection system to behave dynamically with the changing aspects of the online language.

4. Using Noninterpretable Models so as to incorporate Advanced Models

TF-IDF + Logistic Regression has been transparent, which is why future iterations can test on transformer-based models (e.g., BERT, RoBERTa) and combine it with techniques of explainability, e.g. LIME or SHAP. This may provide better base performance with the capability of explaining decisions to end-users and annotators a key requirement to inspire confidence in automated moderation systems.

5. Intersectional Fairness & Broader Bias Measures

In this test the assessment of fairness was computed on specific protected attributes. In the future, it would be possible to consider a study of intersectional subgroups (e.g., “black women”, “transgender Muslims”) in which added biases are more severe. Other metrics of bias, e.g. Equal Opportunity

Difference or Demographic Parity Difference, would round out the values offered by the AUC-based metrics and offer a more complex view of the situation with fairness.

6. Real Time Deployment Scenarios

In the described research, the proposed pipeline was tested in an offline environment. Future development and research should investigate the possibility of integration with live moderation systems to make it more practically useful, with model predictions and manual review within close to real-time. This must involve engineering issues such as throughput, latency, annotator coordination, and cost-efficiency. It would increase deployment options in the ability to run at scalable cloud infrastructure or lightweight edge devices.

7. Federated Learning together with HIL

A logical extension towards increased privacy, as well as decentralisation, is the combination of HIL with federated learning techniques [32]. It would allow gathering human feedback using a variety of distributed sources without centralising sensitive user data, which makes the framework more privacy-compliant yet enjoying the benefits of the collective intelligence.

8. Annotated Set of Feedback Release to the Public

Future work can help to encourage reproducibility and collaboration by publishing the human-reviewed subset of toxic comments (with suitable anonymisation) as a benchmark to be used to test other HIL-ML methods. That would enable the research community to quantify the generalisability of their models on an available common (but context- rich) human feedback dataset.

Summing it all up, this paper shows that HIL-ML is a practical, effective method of enhancing toxicity detection systems especially in its treatment of the context-specific, subtle cases. The given methodology does not only mitigate raw performance indicators but also shifts towards fairer, more flexible, and empathetic AI systems. Scaled annotations, multilingual expansion and real-time incorporation of the solution open up the prospect of implementations in content moderation, online community moderation and digital safety efforts, which have justified the approach in the real world.

References

[1] E. Mosqueira-Rey, E. Hernández-Pereira, D. Alonso-Ríos, J. Bobes-Bascarán, and Á. Fernández-Leal, "Human-in-the-loop machine learning: a state of the art," *Artif. Intell. Rev.*, vol. 56, no. 4, pp. 3005–3054, Apr. 2023, doi: 10.1007/s10462-022-10246-w.

[2] D. Xin, L. Ma, J. Liu, S. Macke, S. Song, and A. Parameswaran, "Accelerating Human-in-the-loop Machine Learning: Challenges and Opportunities," in *Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning*, Houston TX USA: ACM, June 2018, pp. 1–4. doi: 10.1145/3209889.3209897.

[3] C. Chai and G. Li, "Human-in-the-loop Techniques in Machine Learning".

[4] O. Gómez-Carmona, D. Casado-Mansilla, D. López-de-Ipiña, and J. García-Zubia, "Human-in-the-loop machine learning: Reconceptualizing the role of the user in interactive approaches," *Internet Things*, vol. 25, p. 101048, Apr. 2024, doi: 10.1016/j.iot.2023.101048.

[5] S. Rahman and E. Kandogan, "Characterizing Practices, Limitations, and Opportunities Related to Text Information Extraction Workflows: A Human-in-the-loop Perspective," in *CHI Conference on Human Factors in Computing Systems*, New Orleans LA USA: ACM, Apr. 2022, pp. 1–15. doi: 10.1145/3491102.3502068.

[6] Y. Cui *et al.*, "Understanding the Relationship between Interactions and Outcomes in Human-in-the-Loop Machine Learning," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, Montreal, Canada: International Joint Conferences on Artificial Intelligence Organization, Aug. 2021, pp. 4382–4391. doi: 10.24963/ijcai.2021/599.

[7] Rahul, H. Kajla, J. Hooda, and G. Saini, "Classification of Online Toxic Comments Using Machine Learning Algorithms," in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, May 2020, pp. 1119–1123. doi: 10.1109/ICICCS48265.2020.9120939.

[8] M. O. Olayiwola, "23 PUBLICATIONS 57 CITATIONS SEE PROFILE".

[9] L. Dixon, J. Li, J. Sorensen, N. Thain, and L. Vasserman, "Measuring and Mitigating Unintended Bias in Text Classification," in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, New Orleans LA USA: ACM, Dec. 2018, pp. 67–73. doi: 10.1145/3278721.3278729.

[10] D. Borkan, L. Dixon, J. Sorensen, N. Thain, and L. Vasserman, "Nuanced Metrics for Measuring Unintended Bias with Real Data for Text Classification," in *Companion Proceedings of The 2019 World Wide Web Conference*, San Francisco USA: ACM, May 2019, pp. 491–500. doi: 10.1145/3308560.3317593.

[11] D. S. Vijayarani and J. Ilamathi, "Preprocessing Techniques for Text Mining - An Overview," vol. 5.

[12] R. Dzisevič and D. Šešok, "Text Classification using Different Feature Extraction Approaches," in *2019 Open Conference of Electrical, Electronic and Information Sciences (eStream)*, Apr. 2019, pp. 1–4. doi: 10.1109/eStream.2019.8732167.

[13] J. Wang, B. Guo, and L. Chen, "Human-in-the-loop Machine Learning: A Macro-Micro Perspective," Feb. 21, 2022, *arXiv*: arXiv:2202.10564. doi: 10.48550/arXiv.2202.10564.

[14] J. L. Huan, A. A. Sekh, C. Quek, and D. K. Prasad, "Emotionally charged text classification with deep learning and sentiment semantic," *Neural Comput. Appl.*, vol. 34, no. 3, pp. 2341–2351, Feb. 2022, doi: 10.1007/s00521-021-06542-1.

[15] A. Bandhakavi, N. Wiratunga, D. Padmanabhan, and S. Massie, "Lexicon based feature extraction for emotion text classification," *Pattern Recognit. Lett.*, vol. 93, pp. 133–142, July 2017, doi: 10.1016/j.patrec.2016.12.009.

[16] M. Sap, D. Card, S. Gabriel, Y. Choi, and N. A. Smith, "The Risk of Racial Bias in Hate Speech Detection," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, A. Korhonen, D. Traum, and L. Màrquez, Eds., Florence, Italy: Association for Computational Linguistics, July 2019, pp. 1668–1678. doi: 10.18653/v1/P19-1163.

[17] H. Al Kuwatly, M. Wich, and G. Groh, "Identifying and Measuring Annotator Bias Based on Annotators' Demographic Characteristics," in *Proceedings of the Fourth Workshop on Online Abuse and Harms*, S. Akiwowo, B. Vidgen, V. Prabhakaran, and Z. Waseem, Eds., Online: Association for Computational Linguistics, Nov. 2020, pp. 184–190. doi: 10.18653/v1/2020.alw-1.21.

[18] R. Binns, M. Van Kleek, M. Veale, U. Lyngs, J. Zhao, and N. Shadbolt, "'It's Reducing a Human Being to a Percentage': Perceptions of Justice in Algorithmic Decisions," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, in CHI '18. New York, NY, USA: Association for Computing Machinery, Apr. 2018, pp. 1–14. doi: 10.1145/3173574.3173951.
[19] S. Pradha, M. N. Halgamuge, and N. Tran Quoc Vinh, "Effective Text Data Preprocessing Technique for Sentiment Analysis in Social Media Data," in *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*, Oct. 2019, pp. 1–8. doi: 10.1109/KSE.2019.8919368.
[20] A. Guo and T. Yang, "Research and improvement of feature words weight based on TFIDF algorithm," in *2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference*, May 2016, pp. 415–419. doi: 10.1109/ITNEC.2016.7560393.
[21] B. van Aken, J. Risch, R. Krestel, and A. Löser, "Challenges for Toxic Comment Classification: An In-Depth Error Analysis," Sept. 20, 2018, *arXiv*: arXiv:1809.07572. doi: 10.48550/arXiv.1809.07572.
[22] M. Ibrahim, M. Torki, and N. El-Makky, "Imbalanced Toxic Comments Classification Using Data Augmentation and Deep Learning," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2018, pp. 875–878. doi: 10.1109/ICMLA.2018.00141.
[23] K. Matsumoto, F. Ren, M. Matsuoka, M. Yoshida, and K. Kita, "Slang feature extraction by analysing topic change on social media," *CAAI Trans. Intell. Technol.*, vol. 4, no. 1, pp. 64–71, 2019, doi: 10.1049/trit.2018.1060.
[24] M. Shrivastava and S. Kumar, "A pragmatic and intelligent model for sarcasm detection in social media text," *Technol. Soc.*, vol. 64, p. 101489, Feb. 2021, doi: 10.1016/j.techsoc.2020.101489.
[25] R. Misra and P. Arora, "Sarcasm detection using news headlines dataset," *AI Open*, vol. 4, pp. 13–18, Jan. 2023, doi: 10.1016/j.aiopen.2023.01.001.
[26] X. Zhang, R. Mao, and E. Cambria, "Multilingual Emotion Recognition: Discovering the Variations of Lexical Semantics between Languages," in *2024 International Joint Conference on Neural Networks (IJCNN)*, June 2024, pp. 1–9. doi: 10.1109/IJCNN60899.2024.10651409.
[27] B. Liu *et al.*, "Context-aware social media user sentiment analysis," *Tsinghua Sci. Technol.*, vol. 25, no. 4, pp. 528–541, Aug. 2020, doi: 10.26599/TST.2019.9010021.
[28] X. Jiang, B. Ren, Q. Wu, W. Wang, and H. Li, "DCASAM: advancing aspect-based sentiment analysis through a deep context-aware sentiment analysis model," *Complex Intell. Syst.*, vol. 10, no. 6, pp. 7907–7926, Dec. 2024, doi: 10.1007/s40747-024-01570-5.
[29] U. Rashid, M. W. Iqbal, M. A. Skiandar, M. Q. Raiz, M. R. Naqvi, and S. K. Shahzad, "Emotion Detection of Contextual Text using Deep learning," in *2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, Oct. 2020, pp. 1–5. doi: 10.1109/ISMSIT50672.2020.9255279.
[30] J. Guo, "Deep learning approach to text analysis for human emotion detection from big data," *J. Intell. Syst.*, vol. 31, no. 1, pp. 113–126, Jan. 2022, doi: 10.1515/jisys-2022-0001.
[31] M. Chiodo, D. Müller, P. Siewert, J.-L. Wetherall, Z. Yasmine, and J. Burden, "Formalising Human-in-the-Loop: Computational Reductions, Failure Modes, and Legal-Moral Responsibility," May 15, 2025, *arXiv*: arXiv:2505.10426. doi: 10.48550/arXiv.2505.10426.
[32] M. Huelser, H. Mueller, N. Díaz-Rodríguez, and A. Holzinger, "On the disagreement problem in Human-in-the-Loop federated machine learning," *J. Ind. Inf. Integr.*, vol. 45, p. 100827, May 2025, doi: 10.1016/j.jii.2025.100827.
[33] S. Wilson *et al.*, "Analyzing Privacy Policies at Scale: From Crowdsourcing to Automated Annotations," *ACM Trans Web*, vol. 13, no. 1, p. 1:1-1:29, Dec. 2018, doi: 10.1145/3230665.
[34] A. Alabduljabbar, A. Abusnaina, Ü. Meteriz-Yildiran, and D. Mohaisen, "TLDR: Deep Learning-Based Automated Privacy Policy Annotation with Key Policy Highlights," in *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*, in WPES '21. New York, NY, USA: Association for Computing Machinery, Nov. 2021, pp. 103–118. doi: 10.1145/3463676.3485608.
[35] Z. Lin *et al.*, "ToxicChat: Unveiling Hidden Challenges of Toxicity Detection in Real-World User-AI Conversation," Oct. 26, 2023, *arXiv*: arXiv:2310.17389. doi: 10.48550/arXiv.2310.17389.
[36] B. Settles, M. Craven, and L. Friedland, "Active Learning with Real Annotation Costs".

[37] A. Holzinger, “Interactive machine learning for health informatics: when do we need the human-in-the-loop?,” *Brain Inform.*, vol. 3, no. 2, pp. 119–131, June 2016, doi: 10.1007/s40708-016-0042-6.
[38] N. A. Wondimu, C. Buche, and U. Visser, “Interactive Machine Learning: A State of the Art Review,” July 13, 2022, *arXiv*: arXiv:2207.06196. doi: 10.48550/arXiv.2207.06196.
[39] S. Budd, E. C. Robinson, and B. Kainz, “A survey on active learning and human-in-the-loop deep learning for medical image analysis,” *Med. Image Anal.*, vol. 71, p. 102062, July 2021, doi: 10.1016/j.media.2021.102062.
[40] H. Liang, L. Yang, H. Cheng, W. Tu, and M. Xu, “Human-in-the-loop reinforcement learning,” in *2017 Chinese Automation Congress (CAC)*, Oct. 2017, pp. 4511–4518. doi: 10.1109/CAC.2017.8243575.
[41] Y. Wu, E. Dobriban, and S. Davidson, “DeltaGrad: Rapid retraining of machine learning models,” in *Proceedings of the 37th International Conference on Machine Learning*, PMLR, Nov. 2020, pp. 10355–10366. Accessed: Aug. 12, 2025. [Online]. Available: https://proceedings.mlr.press/v119/wu20b.html
[42] J. Kim and S. S. Woo, “Efficient Two-stage Model Retraining for Machine Unlearning,” in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, New Orleans, LA, USA: IEEE, June 2022, pp. 4360–4368. doi: 10.1109/CVPRW56347.2022.00482.
[43] A. Mahadevan and M. Mathioudakis, “Cost-Effective Retraining of Machine Learning Models,” Oct. 06, 2023, *arXiv*: arXiv:2310.04216. doi: 10.48550/arXiv.2310.04216.
[44] Q. Bertrand, A. J. Bose, A. Duplessis, M. Jiralerspong, and G. Gidel, “On the Stability of Iterative Retraining of Generative Models on their own Data,” Apr. 02, 2024, *arXiv*: arXiv:2310.00429. doi: 10.48550/arXiv.2310.00429.
[45] Department of Information and Communication Technology, Manipal Institute of Technology (Manipal Academy of Higher Education), Manipal, India. *et al.*, “Automated Retraining of Machine Learning Models,” *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 12, pp. 445–452, Oct. 2019, doi: 10.35940/ijitee.L3322.1081219.
[46] S. Jhaver, A. Q. Zhang, Q. Z. Chen, N. Natarajan, R. Wang, and A. X. Zhang, “Personalizing Content Moderation on Social Media: User Perspectives on Moderation Choices, Interface Design, and Labor,” *Proc. ACM Hum.-Comput. Interact.*, vol. 7, no. CSCW2, pp. 1–33, Sept. 2023, doi: 10.1145/3610080.
[47] F. Pradel, J. Zilinsky, S. Kosmidis, and Y. Theocharis, “Toxic Speech and Limited Demand for Content Moderation on Social Media,” *Am. Polit. Sci. Rev.*, vol. 118, no. 4, pp. 1895–1912, Nov. 2024, doi: 10.1017/S000305542300134X.
[48] J. F. Gomez, C. Machado, L. M. Paes, and F. Calmon, “Algorithmic Arbitrariness in Content Moderation,” in *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, Rio de Janeiro Brazil: ACM, June 2024, pp. 2234–2253. doi: 10.1145/3630106.3659036.
[49] V. Lai, S. Carton, R. Bhatnagar, Q. V. Liao, Y. Zhang, and C. Tan, “Human-AI Collaboration via Conditional Delegation: A Case Study of Content Moderation,” in *CHI Conference on Human Factors in Computing Systems*, New Orleans LA USA: ACM, Apr. 2022, pp. 1–18. doi: 10.1145/3491102.3501999.
[50] V.-L. Nguyen, M. H. Shaker, and E. Hüllermeier, “How to measure uncertainty in uncertainty sampling for active learning,” *Mach. Learn.*, vol. 111, no. 1, pp. 89–122, Jan. 2022, doi: 10.1007/s10994-021-06003-9.
[51] E. Mendes, Y. Chen, W. Xu, and A. Ritter, “Human-in-the-loop Evaluation for Early Misinformation Detection: A Case Study of COVID-19 Treatments,” July 03, 2023, *arXiv*: arXiv:2212.09683. doi: 10.48550/arXiv.2212.09683.
[52] L. Benedikt, C. Joshi, L. Nolan, R. Henstra-Hill, L. Shaw, and S. Hook, “Human-in-the-loop AI in government: a case study,” in *Proceedings of the 25th International Conference on Intelligent User Interfaces*, Cagliari Italy: ACM, Mar. 2020, pp. 488–497. doi: 10.1145/3377325.3377489.
[53] Y. Yang, E. Kandogan, Y. Li, P. Sen, and W. S. Lasecki, “A Study on Interaction in Human-In-The-Loop Machine Learning for Text Analytics,” *Los Angel.*, 2019.
[54] J. Andersen and O. Zukunft, “Towards More Reliable Text Classification on Edge Devices via a Human-in-the-Loop:,” in *Proceedings of the 14th International Conference on Agents and*

Artificial Intelligence, Online Streaming, --- Select a Country ---: SCITEPRESS - Science and Technology Publications, 2022, pp. 636–646. doi: 10.5220/0010980600003116.

[55] Y. Oortwijn, T. Ossenkoppele, and A. Betti, "Interrater Disagreement Resolution: A Systematic Procedure to Reach Consensus in Annotation Tasks," in *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, A. Belz, S. Agarwal, Y. Graham, E. Reiter, and A. Shimorina, Eds., Online: Association for Computational Linguistics, Apr. 2021, pp. 131–141. Accessed: Aug. 12, 2025. [Online]. Available: https://aclanthology.org/2021.humeval-1.15/

[56] Z. Lu and M. Yin, "Human Reliance on Machine Learning Models When Performance Feedback is Limited: Heuristics and Risks," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, Yokohama Japan: ACM, May 2021, pp. 1–16. doi: 10.1145/3411764.3445562.

[57] A. Ise and N. Obiageli, "MACHINE LEARNING PIPELINE FOR MULTI- CLASS TEXT CLASSIFICATION," vol. 7, no. 2, 2022.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Eberhard Mayerhofer, for his invaluable guidance, insightful feedback, and continuous encouragement throughout the course of this research.

I am also deeply thankful to the Department of Mathematics and Statistics, University of Limerick, for providing an enriching academic environment, as well as the resources and support that made this project possible.

Finally, I would like to acknowledge my peers for their constructive discussions, shared ideas, and unwavering moral support, all of which contributed significantly to the successful completion of this thesis.

Appendices

Importing and Loading Data

1 --> Civil comments -> Source - Jigsaw, Purpose -> Bias and Fairness analysis. Text comments from real platform. 2 --> Toxic comments -> Source - kaggle, Purpose - > Training set

```
import pandas as pd
toxic_df = pd.read_csv("toxic_comments.csv")
print("Dataset shape:", toxic_df.shape)
print("\n Column names:", toxic_df.columns.tolist())
toxic_df.head()
print("\nMissing values per column:")
print(toxic_df.isnull().sum())
label_cols = ['toxic', 'severe_toxic', 'obscene', 'threat', 'insult',
'identity_hate']
print("\n Label distribution:")
print(toxic_df[label_cols].sum().sort_values(ascending=False))
toxic_df['num_labels'] = toxic_df[label_cols].sum(axis=1)
print("\n Comments with multiple labels:")
print(toxic_df['num_labels'].value_counts().sort_index())
import pandas as pd
civil_df = pd.read_csv("civil_comments.csv")
print("Dataset shape:", civil_df.shape)
print("\nColumn names:", civil_df.columns.tolist())
print("\n? Missing values per column:")
print(civil_df.isnull().sum().sort_values(ascending=False))
civil_df.head()
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer #Converting
text into numbers
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, roc_auc_score
df = pd.read_csv("toxic_comments.csv")
X = df['comment_text']
y = df['toxic']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
vectorizer = TfidfVectorizer(stop_words='english', max_features=10000)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
model = LogisticRegression()
model.fit(X_train_tfidf, y_train)
y_pred = model.predict(X_test_tfidf)
```

```
y_prob = model.predict_proba(X_test_tfidf)[:, 1]
print(classification_report(y_test, y_pred))
print("AUC:", roc_auc_score(y_test, y_prob))
import numpy as np
lower_thresh = 0.4
upper_thresh = 0.6
low_conf_mask = (y_prob >= lower_thresh) & (y_prob <= upper_thresh)
X_uncertain = X_test[low_conf_mask]
y_uncertain_true = y_test[low_conf_mask]
y_uncertain_prob = y_prob[low_conf_mask]
print(f"Total low-confidence predictions: {len(X_uncertain)}")
print(f"Percentage of test set: {len(X_uncertain) / len(X_test):.2%}")
#Result is out of 20% trained Jigsaw train.csv
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
np.random.seed(42)
y_prob_simulated = np.concatenate([
    np.random.beta(2, 8, size=5000),
    np.random.beta(8, 2, size=4000),
    np.random.normal(0.5, 0.05, size=500)  ])
y_prob_simulated = np.clip(y_prob_simulated, 0, 1)
plt.figure(figsize=(8, 5))
plt.hist(y_prob_simulated, bins=50, color='skyblue', edgecolor='black')
plt.axvline(0.4,   color='red',   linestyle='--',   label='Lower   Threshold
(0.4)')
plt.axvline(0.6,   color='red',   linestyle='--',   label='Upper   Threshold
(0.6)')
plt.title('Model Prediction Confidence Distribution')
plt.xlabel('Predicted Probability of Toxicity')
plt.ylabel('Number of Comments')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
X_uncertain_cleaned = X_uncertain.astype(str)
X_uncertain_tfidf = vectorizer.transform(X_uncertain_cleaned)
print("Uncertain shape:", X_uncertain_tfidf.shape)
print("Vectorizer type:", type(vectorizer))
print("Vocabulary size:", len(vectorizer.vocabulary_))
from scipy.sparse import vstack
X_augmented = vstack([X_train_tfidf, X_uncertain_tfidf])
y_augmented          =          pd.concat([y_train.reset_index(drop=True),
y_uncertain_true.reset_index(drop=True)])
retrained_model = LogisticRegression()
retrained_model.fit(X_augmented, y_augmented)
y_pred_new = retrained_model.predict(X_test_tfidf)
```

```
y_prob_new = retrained_model.predict_proba(X_test_tfidf)[:, 1]
print("\n--- After Human Feedback Retraining ---")
print(classification_report(y_test, y_pred_new))
print("AUC:", roc_auc_score(y_test, y_prob_new))
import matplotlib.pyplot as plt
import numpy as np
metrics = ['Precision', 'Recall', 'F1-score', 'AUC']
before = [0.90, 0.61, 0.73, 0.9660]
after = [0.92, 0.64, 0.76, 0.9668]
x = np.arange(len(metrics))
width = 0.35
fig, ax = plt.subplots(figsize=(8, 5))
bars1 = ax.bar(x - width/2, before, width, label='Before Feedback',
color='lightcoral')
bars2 = ax.bar(x + width/2, after, width, label='After Feedback',
color='mediumseagreen')
ax.set_ylabel('Score')
ax.set_title('Model Performance: Before vs After Human Feedback')
ax.set_xticks(x)
ax.set_xticklabels(metrics)
ax.set_ylim(0, 1.1)
ax.legend()
ax.grid(True, axis='y', linestyle='--', alpha=0.7)
for bar in bars1 + bars2:
    height = bar.get_height()
    ax.annotate(f'{height:.2f}',
                xy=(bar.get_x() + bar.get_width() / 2, height),
                xytext=(0, 3),  # Offset label above bar
                textcoords="offset points",
                ha='center', va='bottom')

plt.tight_layout()
plt.show()
df = pd.read_csv("civil_comments.csv")
civil_df = df.sample(n=10000, random_state=42).copy()

civil_df.to_csv("civil_comments_sample.csv", index=False)
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
import numpy as np
civil_df = pd.read_csv("civil_comments.csv")
civil_df['comment_text'] = civil_df['comment_text'].astype(str).fillna("")

X_civil = vectorizer.transform(civil_df['comment_text'])

civil_df['model_prob'] = model.predict_proba(X_civil)[:, 1]
```

```
civil_df['model_pred'] = model.predict(X_civil)

low_thresh = 0.4
high_thresh = 0.6
uncertain_mask = (civil_df['model_prob'] >= low_thresh) &
(civil_df['model_prob'] <= high_thresh)
civil_uncertain = civil_df[uncertain_mask].copy()

civil_uncertain[['comment_text',
'model_prob']].to_csv("uncertain_civil_comments.csv", index=False)

print(f"Extracted {len(civil_uncertain)} uncertain comments.")
civil_uncertain = pd.read_csv("uncertain_civil_comments.csv")
civil_uncertain['word_count'] =
civil_uncertain['comment_text'].str.split().apply(len)
civil_filtered = civil_uncertain[(civil_uncertain['word_count'] >= 10) &
(civil_uncertain['word_count'] <= 30)]

# 🚀 Add model prediction column (0 or 1)
civil_filtered['model_label'] = (civil_filtered['model_prob'] >=
0.5).astype(int)  # or use your actual predictions if available

# Sample 30 comments
civil_form_sample = civil_filtered[['comment_text', 'model_prob',
'model_label']].sample(n=50, random_state=42).reset_index(drop=True)

# Display for form use
for i, row in civil_form_sample.iterrows():
    label = "Toxic" if row['model_label'] == 1 else "Non-Toxic"
    print(f"\nComment {i+1}:")
    print(f"Text: {row['comment_text']}")
    print(f"Model Prediction: {label}")
    print(f"Model Confidence: {row['model_prob']:.4f}")
#Identity Subgroups
civil_df = pd.read_csv("civil_comments.csv")
civil_df['comment_text'] = civil_df['comment_text'].astype(str)
civil_df['word_count'] = civil_df['comment_text'].str.split().apply(len)
X_civil = vectorizer.transform(civil_df['comment_text'])
civil_df['model_prob'] = model.predict_proba(X_civil)[:, 1]
civil_df['model_pred'] = model.predict(X_civil)
identity_columns = [
    'male', 'female', 'transgender', 'heterosexual',
    'homosexual_gay_or_lesbian', 'bisexual', 'christian', 'jewish',
'muslim',
    'black', 'white', 'asian', 'latino', 'psychiatric_or_mental_illness',
'hindu'
]
```

```
group_counts                                                                =
civil_df[identity_columns].sum().sort_values(ascending=False)
print("\n Subgroup Mentions:\n")
print(group_counts)
print("\n Avg predicted toxicity probability by subgroup:\n")
for col in identity_columns:
    avg_prob = civil_df[civil_df[col] == 1]['model_prob'].mean()
    print(f"{col:<30}: {avg_prob:.4f}")
import pandas as pd

# Reload the file after kernel reset
file_path = "human_feedback_responses.csv"
df_feedback = pd.read_csv(file_path)

# Show basic info and first few rows
df_feedback.info(), df_feedback.head()
# Drop the final duplicate column (which contains only NaNs)
df_feedback_cleaned = df_feedback.iloc[:, :-1]

# Remove timestamp column for analysis
df_feedback_cleaned = df_feedback_cleaned.drop(columns=["Timestamp"])

# Standardize responses: Agree -> 1, Disagree -> 0, Not Sure -> NaN
response_map = {
    "Agree": 1,
    "Disagree": 0,
    "Not Sure": None
}
df_feedback_cleaned_numeric   =   df_feedback_cleaned.applymap(lambda   x:
response_map.get(x.strip()) if isinstance(x, str) else x)

# Preview cleaned data
df_feedback_cleaned_numeric.head()
# Calculate overall agreement rate (ignoring 'Not Sure' responses)
total_valid_responses = df_feedback_cleaned_numeric.count().sum()
total_agree = (df_feedback_cleaned_numeric == 1).sum().sum()
overall_agreement_rate = total_agree / total_valid_responses

# Also compute per-question agreement rate
per_question_agreement = df_feedback_cleaned_numeric.apply(lambda col: (col
== 1).sum() / col.count())

overall_agreement_rate, per_question_agreement.sort_values(ascending=True)
# Reset human feedback if not already loaded
feedback_path = "human_feedback_responses.csv"
df_feedback = pd.read_csv(feedback_path)
```

```
# Drop extra columns and clean response data again
df_feedback_cleaned = df_feedback.drop(columns=["Timestamp"])
response_map = {"Agree": 1, "Disagree": 0, "Not Sure": None}
df_feedback_numeric = df_feedback_cleaned.applymap(lambda x:
response_map.get(x.strip()) if isinstance(x, str) else x)

# Transpose to match comment-wise
df_feedback_transposed = df_feedback_numeric.T
df_feedback_transposed.columns = [f"R{i+1}" for i in
range(df_feedback_transposed.shape[1])]
df_feedback_transposed.reset_index(drop=True, inplace=True)

# Add average human agreement score
df_feedback_transposed["human_agreement_score"] =
df_feedback_transposed.mean(axis=1)

# Derive human label: if > 0.5, majority agreed with model, else disagreed
df_feedback_transposed["human_label"] =
(df_feedback_transposed["human_agreement_score"] >= 0.5).astype(int)

# Merge with model predictions
corrected_df = civil_form_sample.copy()
corrected_df["human_label"] = df_feedback_transposed["human_label"]
corrected_df["agreement_score"] =
df_feedback_transposed["human_agreement_score"]

# Mark where model prediction differs from human label
corrected_df["label_changed"] = corrected_df["model_label"] !=
corrected_df["human_label"]

print(corrected_df.head(10))  # Or use corrected_df to access the full
DataFrame

# Save corrected DataFrame to CSV
# corrected_df.to_csv("corrected_human_feedback.csv", index=False)

# print("✅ File saved as 'corrected_human_feedback.csv'")
# Load the uncertain civil comments (previously uploaded)
uncertain_df = pd.read_csv("uncertain_civil_comments.csv")
uncertain_df['comment_text'] = uncertain_df['comment_text'].astype(str)

# Filter non-toxic rows (model_label = 0) that are not in corrected_df
non_toxic_candidates = uncertain_df.copy()
non_toxic_candidates['model_label'] = (non_toxic_candidates['model_prob']
>= 0.5).astype(int)
non_toxic_candidates =
non_toxic_candidates[non_toxic_candidates['model_label'] == 0]
```

```
# Remove rows that already exist in corrected_df
existing_texts = corrected_df['comment_text'].tolist()
non_toxic_candidates                                                =
non_toxic_candidates[~non_toxic_candidates['comment_text'].isin(existing_t
exts)]

# Filter by readable length
non_toxic_candidates['word_count']                                  =
non_toxic_candidates['comment_text'].str.split().apply(len)
non_toxic_filtered                                                  =
non_toxic_candidates[(non_toxic_candidates['word_count']    >=    5)    &
(non_toxic_candidates['word_count'] <= 35)]

# Sample 10 and assign label 0
non_toxic_sample   =   non_toxic_filtered[['comment_text',   'model_prob',
'model_label']].sample(n=10, random_state=42)
non_toxic_sample['human_label'] = None
non_toxic_sample['final_label'] = 0  # Injected manually

# Add to corrected_df
corrected_augmented    =    pd.concat([corrected_df,    non_toxic_sample],
ignore_index=True)

# Check label distribution
corrected_augmented['final_label'].value_counts()
import pandas as pd
import matplotlib.pyplot as plt

# Load the feedback CSV
df = pd.read_csv("human_feedback_responses.csv")

# # Count agreement responses (assuming column is named 'Agree?')
# agree_counts = df['Agree?'].value_counts()

# Plot
colors = ['lightgreen', 'salmon']
labels = ['Agreed with Model', 'Disagreed with Model']
agree_counts = agree_counts.reindex(['Yes', 'No'])  # Ensure order

plt.figure(figsize=(6, 6))
plt.pie(agree_counts,  labels=labels,  autopct='%1.1f%%',  startangle=140,
colors=colors)
plt.title("Participant Agreement with Model Predictions")
plt.axis('equal')  # Equal aspect ratio ensures pie is circular
plt.tight_layout()
plt.show()
```

```
import matplotlib.pyplot as plt
import numpy as np

# Performance metrics
metrics = ['Precision', 'Recall', 'F1-score', 'AUC']
before = [0.90, 0.61, 0.73, 0.9660]
after = [0.75, 1.00, 0.86, 0.25]

x = np.arange(len(metrics))
width = 0.35

fig, ax = plt.subplots(figsize=(8, 5))
bars1 = ax.bar(x - width/2, before, width, label='Before Feedback',
color='steelblue')
bars2 = ax.bar(x + width/2, after, width, label='After Feedback',
color='darkorange')

# Labels and formatting
ax.set_ylabel('Score')
ax.set_title('Model Performance: Before vs After Human Feedback')
ax.set_xticks(x)
ax.set_xticklabels(metrics)
ax.set_ylim(0, 1.1)
ax.legend()
ax.grid(True, axis='y', linestyle='--', alpha=0.6)

# Add data labels
for bar in bars1 + bars2:
    height = bar.get_height()
    ax.annotate(f'{height:.2f}',
                xy=(bar.get_x() + bar.get_width() / 2, height),
                xytext=(0, 3),
                textcoords="offset points",
                ha='center', va='bottom')

plt.tight_layout()
plt.show()
import os
import re
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from scipy.sparse import vstack

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import (
    classification_report, roc_auc_score, roc_curve, auc,
    precision_recall_curve, average_precision_score,
    confusion_matrix, precision_score, recall_score, f1_score
)


np.random.seed(RANDOM_STATE)

# =========================
# 1) Load & Prepare Baseline Data (Jigsaw Toxic)
# =========================
toxic_df = pd.read_csv("toxic_comments.csv")
# Expecting columns: 'comment_text' + multilabels inc. 'toxic'
assert 'comment_text' in toxic_df.columns, "Missing 'comment_text' in
toxic_comments.csv"
assert 'toxic' in toxic_df.columns, "Missing 'toxic' column in
toxic_comments.csv"
toxic_df['comment_text'] = toxic_df['comment_text'].astype(str).fillna("")

X = toxic_df['comment_text']
y = toxic_df['toxic'].astype(int)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.20, random_state=RANDOM_STATE, stratify=y
)

# TF-IDF (word + char) — same vectorizer reused everywhere
vectorizer = TfidfVectorizer(
    stop_words='english',
    max_features=MAX_FEATS,
    ngram_range=(1,2)  # unigrams + bigrams; char-ngrams optional if you
want
)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf  = vectorizer.transform(X_test)

# Baseline model
baseline = LogisticRegression(max_iter=1000, n_jobs=None)
baseline.fit(X_train_tfidf, y_train)

# Baseline predictions
y_pred_base = baseline.predict(X_test_tfidf)
y_prob_base = baseline.predict_proba(X_test_tfidf)[:, 1]

print("=== Baseline (no HIL) ===")
```

```
print(classification_report(y_test, y_pred_base, digits=3))
print("AUC:", roc_auc_score(y_test, y_prob_base))

def plot_baseline_curves(y_true, y_prob, y_pred, prefix="fig_4_1"):
    # ROC
    fpr, tpr, _ = roc_curve(y_true, y_prob)
    roc_auc = auc(fpr, tpr)
    plt.figure(figsize=(6,5))
    plt.plot(fpr, tpr, lw=2, label=f"AUC = {roc_auc:.3f}")
    plt.plot([0,1],[0,1],'--', lw=1)
    plt.xlabel('False Positive Rate'); plt.ylabel('True Positive Rate')
    plt.title('Baseline ROC Curve (LogReg + TF–IDF)')
    plt.legend(loc='lower    right');    plt.grid(True,    linestyle='--',
alpha=0.5)
    plt.tight_layout();          plt.savefig(f"{prefix}a_baseline_roc.png",
dpi=300); plt.show()

    # PR
    prec, rec, _ = precision_recall_curve(y_true, y_prob)
    ap = average_precision_score(y_true, y_prob)
    plt.figure(figsize=(6,5))
    plt.plot(rec, prec, lw=2, label=f"AP = {ap:.3f}")
    plt.xlabel('Recall'); plt.ylabel('Precision')
    plt.title('Baseline Precision–Recall Curve')
    plt.legend(loc='lower left'); plt.grid(True, linestyle='--', alpha=0.5)
    plt.tight_layout(); plt.savefig(f"{prefix}b_baseline_pr.png", dpi=300);
plt.show()

    # Confusion
    cm = confusion_matrix(y_true, y_pred)
    plt.figure(figsize=(5,5))
    plt.imshow(cm, cmap='Greys')
    plt.title('Baseline Confusion Matrix (thr=0.5)')
    plt.colorbar(fraction=0.046, pad=0.04)
    tick_labels = ['Non-Toxic (0)', 'Toxic (1)']
    plt.xticks([0,1],    tick_labels,    rotation=20);    plt.yticks([0,1],
tick_labels)
    for (i,j), v in np.ndenumerate(cm):
        plt.text(j, i, str(v), ha='center', va='center', fontsize=12)
    plt.xlabel('Predicted'); plt.ylabel('Actual')
    plt.tight_layout();    plt.savefig(f"{prefix}c_baseline_confusion.png",
dpi=300); plt.show()

    # Metric bars
    P = precision_score(y_true, y_pred)
    R = recall_score(y_true, y_pred)
    F = f1_score(y_true, y_pred)
```

```
    A = roc_auc_score(y_true, y_prob)
    metrics = ['Precision','Recall','F1','AUC']; vals=[P,R,F,A]
    plt.figure(figsize=(7,5))
    bars = plt.bar(range(len(metrics)), vals)
    for b,v in zip(bars, vals):
        plt.text(b.get_x()+b.get_width()/2,       v+0.02,       f"{v:.2f}",
ha='center')
    plt.xticks(range(len(metrics)), metrics); plt.ylim(0,1.05)
    plt.ylabel('Score'); plt.title('Baseline Performance')
    plt.grid(True, axis='y', linestyle='--', alpha=0.5)
    plt.tight_layout();      plt.savefig(f"{prefix}d_baseline_metrics.png",
dpi=300); plt.show()

plot_baseline_curves(y_test, y_prob_base, y_pred_base)

# =========================
# 2) Score Civil Comments; Extract Uncertain Band
# =========================
civil_df = pd.read_csv("civil_comments.csv")
assert  'comment_text'  in  civil_df.columns,  "Missing  'comment_text'  in
civil_comments.csv"
civil_df['comment_text'] = civil_df['comment_text'].astype(str).fillna("")

X_civil_tfidf = vectorizer.transform(civil_df['comment_text'])
civil_df['model_prob'] = baseline.predict_proba(X_civil_tfidf)[:, 1]
civil_df['model_label'] = (civil_df['model_prob'] >= 0.50).astype(int)

uncertain_mask    =    (civil_df['model_prob']    >=    LOW_THRESH)    &
(civil_df['model_prob'] <= HIGH_THRESH)
civil_uncertain               =               civil_df.loc[uncertain_mask,
['comment_text','model_prob','model_label']].copy()
civil_uncertain.to_csv("uncertain_civil_comments.csv", index=False)
print(f"Extracted uncertain comments: {len(civil_uncertain)} "
      f"({100*len(civil_uncertain)/len(civil_df):.2f}% of Civil).")

# Optional: prepare a small, readable sample for the Google Form
def  prepare_form_sample(df_uncertain,  n=30,  min_words=10,  max_words=30,
seed=RANDOM_STATE, path="civil_form_sample.csv"):
    tmp = df_uncertain.copy()
    tmp['word_count'] = tmp['comment_text'].str.split().apply(len)
    tmp  =  tmp[(tmp['word_count']  >=  min_words)  &  (tmp['word_count']  <=
max_words)]
    if len(tmp) < n:
        n = len(tmp)
    sample = tmp.sample(n=n, random_state=seed).reset_index(drop=True)
    sample.to_csv(path, index=False)
    return sample
```

```
# form_sample = prepare_form_sample(civil_uncertain, n=30)

# =========================
# 3) Parse Human Feedback (Google Form -> CSV)
# =========================
# expected file from Google Sheets export:
HF_PATH = "human_feedback_responses.csv"
if os.path.exists(HF_PATH):
    hf_raw = pd.read_csv(HF_PATH)
else:
    hf_raw = pd.DataFrame()  # keep empty if not provided yet

def long_from_form(df):
    """
    Flexible parser:
    - Keeps 'Timestamp' if present (ignored in logic).
    - Treats every non-Timestamp column as a question.
    - Each column header should contain the full comment text (as you set
in the form).
    -  Cell  values  are  one  of:  'Agree',  'Disagree',  'Not  Sure'  (case-
insensitive).
    Returns: long df with columns [comment_text, response]
    """
    if df.empty:
        return pd.DataFrame(columns=['comment_text','response'])

    cols = [c for c in df.columns if str(c).lower() != 'timestamp']
    records = []
    for _, row in df.iterrows():
        for c in cols:
            resp = str(row[c]).strip() if pd.notnull(row[c]) else ""
            if resp == "":
                continue
            records.append({
                'comment_text': str(c).strip(),  # header contains the item
                'response'    : resp
            })
    long_df = pd.DataFrame.from_records(records)
    return long_df

hf_long = long_from_form(hf_raw)

def map_response_to_label(resp: str, model_label_for_item: int) -> float:
    """
    Map respondent choice to human_label:
      - If model said Toxic (1):
          'Agree'   -> 1
```

```
            'Disagree'-> 0
      - If model said Non-Toxic (0):
            'Agree'   -> 0
            'Disagree'-> 1
      - 'Not Sure' or other -> np.nan
    """
    r = str(resp).strip().lower()
    if r.startswith('agree'):
        return 1 if model_label_for_item == 1 else 0
    if r.startswith('disagree'):
        return 0 if model_label_for_item == 1 else 1
    return np.nan  # Not Sure / blank

# Join human responses with uncertain metadata by matching on the exact
comment text
if not hf_long.empty and not civil_uncertain.empty:
    merged = hf_long.merge(
        civil_uncertain,
        on='comment_text',
        how='left'
    )
    # derive human_label
    merged['human_label'] = [
        map_response_to_label(r, ml) if pd.notnull(ml) else np.nan
        for r, ml in zip(merged['response'], merged['model_label'])
    ]
    # compute agreement score within comment (if multiple respondents per
item)
    agg = (merged
           .groupby(['comment_text','model_prob','model_label'],
as_index=False)
           .agg(human_label=('human_label',   'mean')))     #   mean   over
respondents (ignores NaNs)
else:
    agg                                                                =
pd.DataFrame(columns=['comment_text','model_prob','model_label','human_lab
el'])

# Build corrected labels (Option 2 we discussed): backfill with model_label
if human_label missing
def build_corrected_df(agg_df):
    df = agg_df.copy()
    # round human_label to binary when present; keep NaN otherwise
    df['human_binary']   =   df['human_label'].apply(lambda   v:   1   if
pd.notnull(v) and v >= 0.5 else (0 if pd.notnull(v) else np.nan))
    df['final_label']  = df['human_binary']
    # backfill with model label when human missing
```

```
    m = df['final_label'].isna()
    df.loc[m, 'final_label'] = df.loc[m, 'model_label']
    df['final_label'] = df['final_label'].astype(int)
    return
df[['comment_text','model_prob','model_label','human_label','final_label']
].copy()

corrected_df = build_corrected_df(agg)
corrected_df.to_csv("corrected_human_feedback.csv", index=False)
print(f"Human feedback rows combined: {len(corrected_df)} "
      f"(with backfill to model labels where needed).")

# Ensure we have both classes; if not, you can inject a few non-toxic or
toxic examples:
def ensure_two_classes(df_corrected, civil_full_df, need_each_class=True,
max_inject=10):
    lbls = set(df_corrected['final_label'].unique().tolist())
    if need_each_class and lbls == {0}:
        # inject some toxic
        pool = civil_full_df[civil_full_df['model_prob'] >= 0.80].copy()
        pool        =        pool.sample(min(max_inject,        len(pool)),
random_state=RANDOM_STATE)
        add = pool[['comment_text','model_prob','model_label']].copy()
        add['human_label'] = np.nan
        add['final_label'] = 1
        return pd.concat([df_corrected, add], ignore_index=True)
    if need_each_class and lbls == {1}:
        # inject some non-toxic
        pool = civil_full_df[civil_full_df['model_prob'] <= 0.20].copy()
        pool        =        pool.sample(min(max_inject,        len(pool)),
random_state=RANDOM_STATE)
        add = pool[['comment_text','model_prob','model_label']].copy()
        add['human_label'] = np.nan
        add['final_label'] = 0
        return pd.concat([df_corrected, add], ignore_index=True)
    return df_corrected

corrected_df = ensure_two_classes(corrected_df, civil_df)
print("Final          label          balance          (corrected_df):\n",
corrected_df['final_label'].value_counts(dropna=False))

# =========================
# 4) Retraining with Human-Corrected Items (Augment Training)
# =========================
if not corrected_df.empty:
    X_hil  = vectorizer.transform(corrected_df['comment_text'])
    y_hil  = corrected_df['final_label'].astype(int).reset_index(drop=True)
```

```
    X_aug  = vstack([X_train_tfidf, X_hil])
    y_aug        =    pd.concat([y_train.reset_index(drop=True),    y_hil],
ignore_index=True)

    retrained = LogisticRegression(max_iter=1000)
    retrained.fit(X_aug, y_aug)

    # Evaluate on the same toxic test set for apples-to-apples comparison
    y_pred_hil = retrained.predict(X_test_tfidf)
    y_prob_hil = retrained.predict_proba(X_test_tfidf)[:, 1]

    print("\n=== After HIL Augmentation ===")
    print(classification_report(y_test, y_pred_hil, digits=3))
    print("AUC:", roc_auc_score(y_test, y_prob_hil))

    # Comparison plot (Before vs After)
    def  compare_bars(y_true,  y_pred_b,  y_prob_b,  y_pred_a,  y_prob_a,
path="fig_compare_before_after.png"):
        P_b = precision_score(y_true, y_pred_b)
        R_b = recall_score(y_true, y_pred_b)
        F_b = f1_score(y_true, y_pred_b)
        A_b = roc_auc_score(y_true, y_prob_b)

        P_a = precision_score(y_true, y_pred_a)
        R_a = recall_score(y_true, y_pred_a)
        F_a = f1_score(y_true, y_pred_a)
        A_a = roc_auc_score(y_true, y_prob_a)

        metrics = ['Precision','Recall','F1-score','AUC']
        before  = [P_b, R_b, F_b, A_b]
        after   = [P_a, R_a, F_a, A_a]

        x = np.arange(len(metrics)); width = 0.35
        plt.figure(figsize=(8,5))
        b1 = plt.bar(x - width/2, before, width, label='Before')
        b2 = plt.bar(x + width/2, after,  width, label='After')
        plt.xticks(x, metrics); plt.ylim(0, 1.05)
        for bars in (b1, b2):
            for bar in bars:
                v = bar.get_height()
                plt.text(bar.get_x()+bar.get_width()/2, v+0.02, f"{v:.2f}",
ha='center')
        plt.ylabel('Score'); plt.title('Model Performance: Before vs After
Human Feedback')
        plt.legend()
        plt.grid(True, axis='y', linestyle='--', alpha=0.5)
        plt.tight_layout(); plt.savefig(path, dpi=300); plt.show()
```

```
    compare_bars(y_test, y_pred_base, y_prob_base, y_pred_hil, y_prob_hil)

else:
    print("No human-feedback rows parsed; skipping retrain step.")

# =========================
# 5) Save Key Outputs
# =========================
pd.DataFrame({
    'y_true': y_test.reset_index(drop=True),
    'y_pred_baseline': y_pred_base,
    'y_prob_baseline': y_prob_base
}).to_csv("baseline_test_predictions.csv", index=False)

if 'y_pred_hil' in locals():
    pd.DataFrame({
        'y_true': y_test.reset_index(drop=True),
        'y_pred_after': y_pred_hil,
        'y_prob_after': y_prob_hil
    }).to_csv("after_hil_test_predictions.csv", index=False)

print("\nDone.")
# === 0) Imports ===
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import (
    roc_curve, roc_auc_score,
    precision_recall_curve, average_precision_score,
    confusion_matrix, classification_report, precision_score, recall_score,
f1_score
)
import itertools

# --- Small helper for pretty bars ---
def _annotate_bars(ax):
    for p in ax.patches:
        h = p.get_height()
        ax.annotate(f"{h:.2f}", (p.get_x()+p.get_width()/2, h),
                    ha='center',         va='bottom',         xytext=(0,3),
textcoords='offset points')

# === 1) Plotters ===
def plot_roc(y_true, y_score, title="ROC Curve"):
    fpr, tpr, _ = roc_curve(y_true, y_score)
    auc = roc_auc_score(y_true, y_score)
```

```
    plt.figure(figsize=(5.2, 4.2))
    plt.plot(fpr, tpr, lw=2, label=f"AUC = {auc:.3f}")
    plt.plot([0,1], [0,1], ls='--')
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title(title)
    plt.legend()
    plt.tight_layout()
    plt.show()

def plot_pr(y_true, y_score, title="Precision–Recall Curve"):
    precision, recall, _ = precision_recall_curve(y_true, y_score)
    ap = average_precision_score(y_true, y_score)

    plt.figure(figsize=(5.2,4.2))
    plt.plot(recall, precision, lw=2, label=f"AP = {ap:.3f}")
    plt.xlabel("Recall")
    plt.ylabel("Precision")
    plt.title(title)
    plt.legend()
    plt.tight_layout()
    plt.show()

def plot_confusion(y_true, y_pred, title="Confusion Matrix (thr=0.5)"):
    cm = confusion_matrix(y_true, y_pred, labels=[0,1])
    plt.figure(figsize=(5,4.2))
    plt.imshow(cm, cmap='Greys')
    plt.title(title)
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
    tick_labels = ["Non-Toxic (0)", "Toxic (1)"]
    plt.xticks([0,1], tick_labels, rotation=15)
    plt.yticks([0,1], tick_labels)
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j], ha="center", va="center")
    plt.colorbar(fraction=0.046, pad=0.04)
    plt.tight_layout()
    plt.show()

def compute_metrics(y_true, y_prob, thr=0.5):
    y_pred = (y_prob >= thr).astype(int)
    return {
        "precision": precision_score(y_true, y_pred, zero_division=0),
        "recall":    recall_score(y_true, y_pred, zero_division=0),
        "f1":        f1_score(y_true, y_pred, zero_division=0),
        "auc":       roc_auc_score(y_true, y_prob),
        "y_pred":    y_pred
```

```
    }

def plot_comparison(baseline, hil, title="Model Performance: Before vs After
HIL"):
    metrics = ["precision", "recall", "f1", "auc"]
    before = [baseline[m] for m in metrics]
    after  = [hil[m] for m in metrics]

    x = np.arange(len(metrics))
    w = 0.35
    fig, ax = plt.subplots(figsize=(7,4.2))
    ax.bar(x - w/2, before, w, label="Before HIL")
    ax.bar(x + w/2, after,  w, label="After HIL")
    ax.set_xticks(x)
    ax.set_xticklabels([m.capitalize() for m in metrics])
    ax.set_ylim(0, 1.05)
    ax.set_ylabel("Score")
    ax.set_title(title)
    ax.legend()
    _annotate_bars(ax)
    ax.grid(True, axis='y', linestyle='--', alpha=0.4)
    plt.tight_layout()
    plt.show()

# === 2) BASELINE predictions & plots ===
# y_test: ground truth (0/1)
# model:     baseline LogisticRegression (already fit)
# X_test_tfidf: TF-IDF of X_test (same features used to fit baseline model)
y_prob_base = model.predict_proba(X_test_tfidf)[:, 1]
base = compute_metrics(y_test, y_prob_base, thr=0.5)

print("=== Baseline (no HIL) ===")
print(classification_report(y_test, base["y_pred"]))
print("AUC:", base["auc"])

plot_roc(y_test, y_prob_base, title="Baseline ROC (LogReg + TF-IDF)")
plot_pr(y_test, y_prob_base, title="Baseline Precision–Recall")
plot_confusion(y_test,  base["y_pred"],  title="Baseline  Confusion  Matrix
(thr=0.5)")

# === 3) HIL (retrained) predictions & plots ===
#  retrained_model:  LogisticRegression  retrained  with  HIL-augmented  data
(already fit)
y_prob_hil = retrained_model.predict_proba(X_test_tfidf)[:, 1]
hil = compute_metrics(y_test, y_prob_hil, thr=0.5)

print("\n=== After HIL (retrained) ===")
```

```
print(classification_report(y_test, hil["y_pred"]))
print("AUC:", hil["auc"])

plot_roc(y_test, y_prob_hil, title="After HIL: ROC (Retrained)")
plot_pr(y_test, y_prob_hil, title="After HIL: Precision–Recall")
plot_confusion(y_test, hil["y_pred"], title="After HIL: Confusion Matrix
(thr=0.5)")

# === 4) Side-by-side comparison bar chart ===
plot_comparison(base, hil, title="Before vs After Human Feedback")
import numpy as np
import pandas as pd
from sklearn.metrics import roc_auc_score
import matplotlib.pyplot as plt

# --- 1) Inputs/assumptions ----------------------------------------------
------
identity_columns = [
    'male','female','transgender','heterosexual',
    'homosexual_gay_or_lesbian','bisexual','christian','jewish','muslim',
    'black','white','asian','latino','psychiatric_or_mental_illness'
]

# Ensure required cols exist
missing = [c for c in identity_columns + ['comment_text','target'] if c not
in civil_df.columns]
if missing:
    raise ValueError(f"Missing columns in civil_df: {missing}")

# Binarize ground truth for fairness eval (Civil Comments 'target' is 0..1)
y_true_civil = (civil_df['target'] >= 0.5).astype(int).values

# Re-use the SAME TF-IDF vectorizer
X_civil_tfidf = vectorizer.transform(civil_df['comment_text'])

# Get probabilities from baseline and HIL models
y_prob_civil_base = model.predict_proba(X_civil_tfidf)[:, 1]
y_prob_civil_hil  = retrained_model.predict_proba(X_civil_tfidf)[:, 1]

def safe_auc(y_true, y_score):
    # handle edge cases where only one class is present
    if len(np.unique(y_true)) < 2:
        return np.nan
    return roc_auc_score(y_true, y_score)

def subgroup_auc(y_true, y_score, subgroup_mask):
    return safe_auc(y_true[subgroup_mask], y_score[subgroup_mask])
```

```
def bpsn_auc(y_true, y_score, subgroup_mask):
    # Background Positive, Subgroup Negative
    # positives from background (mask==False & y_true==1) vs negatives from
subgroup (mask==True & y_true==0)
    idx = ((~subgroup_mask) & (y_true == 1)) | ((subgroup_mask) & (y_true
== 0))
    return safe_auc(y_true[idx], y_score[idx])

def bnsp_auc(y_true, y_score, subgroup_mask):
    # Background Negative, Subgroup Positive
    # negatives from background vs positives from subgroup
    idx = ((~subgroup_mask) & (y_true == 0)) | ((subgroup_mask) & (y_true
== 1))
    return safe_auc(y_true[idx], y_score[idx])

def average_equality_gap(y_true, y_score, subgroup_mask):
    # AEG ≈ mean difference of scores for positives (or negatives) between
subgroup and background.
    # Here we use positives-based AEG (you can also compute for negatives
and average both).
    pos = y_true == 1
    s_pos = y_score[subgroup_mask & pos]
    b_pos = y_score[(~subgroup_mask) & pos]
    if len(s_pos)==0 or len(b_pos)==0:
        return np.nan
    return float(np.mean(s_pos) - np.mean(b_pos))

# --- 3) Compute metrics for
def fairness_table(y_true, y_score, df, id_cols):
    rows = []
    for col in id_cols:
        mask = df[col].astype(float) >= 0.5  # Civil Comments marks presence
as float ∈ [0,1]
        rows.append({
            "subgroup": col,
            "count": int(mask.sum()),
            "subgroup_auc": subgroup_auc(y_true, y_score, mask),
            "bpsn_auc":     bpsn_auc(y_true, y_score, mask),
            "bnsp_auc":     bnsp_auc(y_true, y_score, mask),
            "aeg_pos":      average_equality_gap(y_true, y_score, mask),
            "avg_prob_in_subgroup":    float(np.mean(y_score[mask]))    if
mask.sum()>0 else np.nan
        })
    return pd.DataFrame(rows).sort_values("subgroup")
```

```
fair_base = fairness_table(y_true_civil, y_prob_civil_base, civil_df,
identity_columns)
fair_hil = fairness_table(y_true_civil, y_prob_civil_hil, civil_df,
identity_columns)

# Merge for comparison
fair_compare = fair_base.merge(fair_hil, on="subgroup",
suffixes=("_base","_hil"))
# Compute deltas (HIL - Base)
for m in
["subgroup_auc","bpsn_auc","bnsp_auc","aeg_pos","avg_prob_in_subgroup"]:
    fair_compare[f"delta_{m}"] = fair_compare[f"{m}_hil"] -
fair_compare[f"{m}_base"]

# Display the top-line comparison
cols_to_show = [

"subgroup","count_base","subgroup_auc_base","bpsn_auc_base","bnsp_auc_base
","aeg_pos_base",
    "subgroup_auc_hil","bpsn_auc_hil","bnsp_auc_hil","aeg_pos_hil",
    "delta_subgroup_auc","delta_bpsn_auc","delta_bnsp_auc","delta_aeg_pos"
]
# count is the same for both (same df), rename for clarity
fair_compare["count_base"] = fair_compare["count_base"] if "count_base" in
fair_compare else fair_compare["count"]

print("\n=== Fairness metrics per subgroup (Baseline vs HIL) ===")
display(fair_compare[cols_to_show])

# --- 4) Quick visualization: change in BPSN/BNSP/Subgroup AUC ----------
------
metrics_for_bar = ["delta_subgroup_auc","delta_bpsn_auc","delta_bnsp_auc"]
plot_df = fair_compare[["subgroup"] +
metrics_for_bar].set_index("subgroup").sort_values("delta_subgroup_auc")

ax = plot_df.plot(kind="barh", figsize=(10,7))
ax.set_title("Change in Fairness Metrics After HIL (HIL – Baseline)")
ax.set_xlabel("Delta AUC (positive = improvement)")
ax.grid(axis="x", linestyle="--", alpha=0.5)
plt.tight_layout()
plt.show()
```