Published in International Journal of Data Science and Analytics (JDSA), 22(297):1-24, September 2026. Springer.

A Profit-Driven Simulation Framework for Real-Time Evaluation of Toxicity Detection Models in Social Media

Arezo Bodaghi[1*], Benjamin C. M. Fung[2], Jonathan Shahen[3], Ketra A. Schmitt[4]

[1*]Department of Cybersecurity and Intelligent Systems Engineering, Concordia University, Montreal, H3G 2W1, Quebec, Canada.
[2]School of Information Studies, McGill University, Montreal, H3A 1X1, Quebec, Canada.
[3]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, N2L 3G1, Ontario, Canada.
[4]Centre for Engineering in Society, Concordia University, Montreal, H3G 1M8, Quebec, Canada.

*Corresponding author(s). E-mail(s): arezo.bodaghi@concordia.ca;
Contributing authors: ben.fung@mcgill.ca;
jonathan.shahen@scrawlr.com; ketra.schmitt@concordia.ca;

Abstract

Real-time detection of toxic comments on social media is essential for maintaining safe and inclusive online spaces. However, most existing AI moderation systems are evaluated in static, offline environments using standard accuracy metrics such as F1-score, which fail to reflect the complexities of real-world social platforms. These traditional evaluations overlook critical operational factors, including processing speed, user engagement, and the varying sensitivity of different users to harmful content. Moreover, social media environments are diverse ranging from low-toxicity personal networks to high-toxicity public spaces, and a one-size-fits-all model may not perform optimally across all scenarios. To address these limitations, we introduce a novel Profit-Driven Simulation (PDS) framework that evaluates the real-time performance of toxicity detection classifiers under realistic deployment conditions. Developed in collaboration with an industry partner, our framework simulates user behavior, comment flow, and engagement patterns to assess each model's effectiveness, efficiency, and impact on platform profitability.

We tested eight deep learning classifiers including CNN, CNN-FastText, BERT, RoBERTa, and their lightweight variants using a week-long simulation involving 10,000 users across environments with varying levels of toxicity. Results show that no single model is optimal for all contexts: high-throughput models perform best in low- and high-toxicity settings, while moderately accurate models with reasonable efficiency excel in medium-toxicity environments. These findings highlight the need to move beyond accuracy-focused evaluation and adopt context-aware, operationally grounded approaches when selecting classifiers for real-world deployment. The PDS framework offers a practical, scalable approach for evaluating content moderation systems under configurable assumptions

Keywords: Toxicity Detection, Real-Time Content Moderation, Context-Aware Model Evaluation, Simulation Framework, Classifier Efficiency, Classifier Selection

1 Introduction

Antisocial behavior on social media platforms remains a serious challenge. It disproportionately harms marginalized communities and often drives users away from online spaces [1–4]. Toxic comments whether in the form of hate speech, harassment, or direct threats not only discourage participation but also erode the quality of public conversation [5–7]. For many users, repeated exposure to this content leads to self-censorship, withdrawal, and a sense that online environments are unsafe [8]. In response, platforms have promised to improve their moderation practices, with a particular focus on protecting groups that face higher risks, such as women [9].

In recent years, deep learning (DL) methods have become central to automated toxicity detection. These models perform well in controlled experiments, yet they are almost always assessed with accuracy-driven metrics in static, offline settings. Such evaluations do not capture the reality of large, dynamic platforms. Social media continues to expand rapidly: by 2025 more than 5.42 billion people are active online [10], and platforms such as X (formerly Twitter) and Facebook each report hundreds of millions of users producing enormous volumes of content every day [11, 12]. Handling this content stream in real time is far from trivial, given its speed, variety, and scale [13].

Although DL models are effective at flagging toxic material [14, 15], deploying them in live environments raises other concerns. Real-time moderation at scale is computationally demanding, and traditional evaluation rarely takes into account latency, throughput, or moderation costs. Users also have little tolerance for delay: they expect posts to appear instantly [16, 17]. Even small lags or errors can reduce trust, harm user experience, and eventually drive people away [18–20].

Another complication is that toxicity is not evenly distributed. The majority of social media content is benign [21, 22], while the harmful fraction is smaller, often context-dependent, and harder to detect reliably. Language that is offensive in one situation may be neutral in another [15, 23]. Patterns of activity also differ across users, with some posting rarely and others flooding conversations [24]. In this environment, a single classifier tuned for accuracy alone is unlikely to capture the complexity of real interactions.

To address these gaps, we propose a Profit-Driven Simulation (PDS) framework. Developed in collaboration with the social media platform Scrawlr, PDS recreates real-time flows of user content and behavior under different levels of toxicity. Unlike most prior work, our framework does not stop at measuring accuracy. It evaluates models along dimensions that matter to platforms in practice: throughput, operational cost, user experience, and overall profitability.

The aim of this study is to identify which classifiers perform best under realistic conditions and which can deliver the greatest value to platforms while protecting users. We focus on three guiding research questions:

- How should toxicity detection models be evaluated under large-scale, real-time conditions where offline accuracy metrics are insufficient?
- How do different classifiers compare in terms of predictive performance, computational efficiency, and operational cost across varying toxicity distributions?
- Can a profit-driven simulation framework identify deployment-optimal models that balance user protection, platform objectives, and resource constraints prior to real-world implementation?

The contributions of this paper are fourfold. First, we introduce a deployment-oriented evaluation paradigm for toxicity detection that integrates predictive performance with operational constraints, including throughput, latency, user sensitivity, and moderation cost. Second, we develop the Profit-Driven Simulation (PDS) framework, which models real-time user activity, content flow, and varying toxicity distributions to approximate realistic social media environments. Third, through extensive experiments involving eight deep learning classifiers, we demonstrate that models optimized for offline accuracy are not necessarily optimal under real-time, large-scale constraints. Our findings reveal context-dependent trade-offs between predictive performance and computational efficiency. Finally, we provide actionable insights for researchers and platform designers, showing how simulation-based evaluation can guide classifier selection prior to costly real-world deployment.

The rest of the paper is organized as follows. Section 2 reviews prior work on toxicity detection. Section 3 introduces the PDS framework and its components. Section 4 describes the experimental design and classifier configurations. Section 5 presents our findings. Section 6 concludes the paper, discusses its limitations, and outlines directions for future work.

2 Background

In this section, we offer a brief overview of the detection of toxic content in social media texts. This will underscore essential aspects of social media platforms, toxic content, and the array of solutions employed to combat this issue.

2.1 Social Media

Microblogging platforms are networks of real-time information, and their true value lies in their ability to absorb timely and relevant information that is significant to users [25]. Real-time platforms promise an immediate experience: users expect their generated content to be visible to followers within seconds, and any prolonged delay could lead to user churn

[16]. The overwhelming flow of content generated by users, who are imagined as potential super processors [26], creates a significant challenge in real-time dissemination, computing, processing, and analysis of data. It is difficult to effectively process big data as a result of its variety, velocity, volume, and value [13]. Therefore, the aspect of real-time processing plays a pivotal role in addressing online toxicity, and the developed models should be able to handle a substantial volume of data within a second. Unfortunately, this aspect has been largely overlooked in the literature. To the best of our knowledge and based on our review of the literature, we did not identify prior work that jointly considers throughput, latency, and cost for the evaluation of toxicity detection on social media platforms.

2.2 Toxicity in Social Media

Although a significant volume of content is shared every second, the majority of it is relatively harmless, nontoxic, and knowledge-based. This leaves only a small fraction that is toxic, often representing attacks on other users, individuals, or minority groups [21, 22]. This difference in rates of toxic and nontoxic content complicates the task of flagging toxic content and increases the risk of false positives.

A dataset provided by [27] consists of 80K tweets, each annotated with five judgments. The dataset is categorized into four groups: Abusive (11%), Hateful (7.5%), Spam (22.5%), and Normal (59%). It reveals that over half of the tweets do not contain any toxic language and are classified as normal. Furthermore, the analysis of 293 million English tweets using Google's Perspective API models by Qayyum et al. [28] demonstrated that 80 percent had toxicity scores below 0.40. In particular, users exhibiting high toxicity are often reported for misconduct violations and subsequently banned [29]. However, the enforcement of rules can sometimes be uneven.

Furthermore, a study on 1.18M Twitter (now X) conversations, encompassing 58.5M tweets and 4.4M users, found that users with moderate activity levels displayed a higher proportion of toxic tweets than both low- and high-activity users [30]. The study conducted in this paper determined that users posting between 10 and 1000 tweets were responsible for 22% to 23% of toxic tweets. On the contrary, highly active users with a substantial number of posts (2000-10000) exhibited a lower percentage of toxic tweets (2%-19%) than low-activity users. Furthermore, more than 50% of the users did not post any toxic tweets. It should be noted that tweets from influential users often attract more replies from users with fewer followers, which increases the likelihood of influential users being targeted with toxic comments.

In another study by Radfar et al. [31], 178K tweets were collected from users with different follower-friend relationships. Radfar et al. found that almost 9.6% shared tweets between users who are mutual friends were deleted due to the high ratio of toxicity. On the contrary, Twitter removed almost 40% of tweets shared between users who are not mutually connected or there is only a one-way connection due to toxicity. They later selected 6.7K tweets and annotated them by human experts. Tweets between users who do not have a connection were found to be nearly three times more likely to be toxic than tweets between mutual friends. Additionally, they also found that users follow great users to read their

posts while targeting them with a toxic message.

Thus, the variety in social media content and users emphasizes the limits of relying on a single classifier to handle all platform situations effectively. Consequently, the dynamic nature of social media platforms demands different techniques for different cases. However, identifying the detector most suitable for various scenarios remains a significant challenge.

2.3 Existing Approaches for Toxicity Detection

Recent years have witnessed a substantial surge in research dedicated to exploring both the social and computational dimensions of toxic content detection [32–34]. Existing research has tackled diverse aspects of online toxicity across various platforms. This includes endeavors such as the detection and categorization of toxic content [35–37], assessment of the impact of online toxicity on communities [38, 39], characterization of different types of toxicity [31], and identification of toxic users [40]. In recent years, machine learning (ML) approaches, including classical ML and DL algorithms, have been implemented mainly to identify toxicity in online conversations [41].

Numerous studies have used various classical ML techniques to tackle online toxicity. For example, logistic regression (LR) [42, 43], decision trees, random forest (RF), and support vector machine (SVM) [40] have been used effectively to identify toxic content. Moreover, approaches such as combining Latent Semantic Analysis (LSA) with SVM and using Latent Dirichlet Allocation (LDA) [44] have been explored to address this challenge. Additionally, ensemble techniques, including gradient boosting, stacking, and bagging, have been successfully applied within this context [36]. In terms of feature extraction, methods such as bag-of-words (BOW), Term-frequency-inverse document frequency (TF-IDF), and word embeddings are commonly employed [45]. The literature offers numerous studies in which different techniques have been compared, considering various feature extraction and pre-processing methods [46, 47].

Since 2006, research has shifted to DL-based models derived from Neural Networks (NN), a branch of ML that is rapidly gaining traction [48]. Various Deep Neural Networks (DNN)-based models have been also used extensively for detecting toxic content, such as Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), Long Short-term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Bidirectional GRU (BiGRU) [49–52]. CNN is a specific type of forward neural network that has a layer of neurons to perform convolution and is one of the most widely used algorithms to classify toxic content [5, 41, 53].

With the introduction of pre-trained word embeddings, recent trends in text classification tasks have switched to using language models that are trained on large unlabeled corpora. The work by Collobert and Weston [54] was the first to show the usefulness of pre-trained word embeddings and established word embeddings as a highly effective tool when used in downstream tasks. Later in 2013, Mikolov et al. [55] created word2vec, followed in 2014 by Global Vectors(GloVe), introduced by Pennington et al. in 2014 [56]. Furthermore, fastText embeddings is a vector representation technique created and released by Facebook AI research [57]. It is trained on Common Crawl

and Wikipedia for 157 languages [58]. As the name implies, it is an efficient and fast method of learning word representations and performing tasks. Rather than learning word representations, fastText embeddings focus on considering the internal structure of words. This is particularly useful for languages with a large number of morphological forms so that learners of these languages can learn representations independently [59]. Word embeddings can be used as input features to DNN classifiers. The study by Mohammed et al. [60] compared CNN, RNN, LSTM, bi-LSTM, GRU, and bi-GRU models in two scenarios. The first utilized a multi-label classification model with a standard embedding layer, while the second integrated pre-trained embedding corpora such as Glove, word2vec, and fastText for these models.

Bidirectional Encoder Representations from Transformers (BERT) is another successful pre-trained language model introduced by Devlin et al. in 2018 [61]. It utilizes transformer-based architecture to comprehend contextual word relationships, generating contextualized word embeddings. Unlike other techniques, BERT's embeddings adapt based on sentence context, enhancing word meaning understanding. This model is pre-trained on extensive text data and fine-tuned for tasks including sentiment analysis and question answering [62]. It employs tokenization and multi-layer bidirectional transformers to encode tokens, benefiting from an attention mechanism that captures inter-token dependencies [63]. There are several variants of BERT that have been developed to improve its efficiency and effectiveness. BERT-base is the original version of BERT that contains 12 transformer layers and 110 million parameters [61]. ALBERT, introduced by Lan et al. in 2020, is a modified version of BERT that uses parameter reduction techniques to achieve better performance with fewer parameters [64]. DistilBERT, introduced by Sanh et al. in 2019 [65], is a distilled version of BERT that is smaller in size and faster in speed, while maintaining comparable performance. In 2019, Liu et al. [66] introduced RoBERTa, which is a large-scale BERT model that uses dynamic masking during pre-training to improve its effectiveness.

Using pre-trained word embeddings has become increasingly common since the introduction of transformer-based structures for downstream text classification. For example, in a study by Zhao et al. [67], the performance of different pre-trained language models was compared using three distinct architectures for toxic language classification: BERT, RoBERTa, XLM, a bi-LSTM + (BERT or RoBERTa or XLM), and a CNN + (BERT or RoBERTa or XLM). Furthermore, the research conducted by [68] involved evaluating the BERT-base model against three counterparts, Multilingual BERT, RoBERTa and DistilBERT. In particular, the BERT-based model outperformed all the others, achieving the highest level of performance.

2.4 Evaluation Metrics

Various metrics have been used to assess algorithm performance and compare different techniques developed to detect toxic language. A systematic review of ML techniques for toxic comment classification [41] indicates that F1-score, accuracy, and area under the ROC curve (AUC ROC) are the most widely used evaluation metrics. Additionally,

other metrics including Log Loss, Hamming Loss, Mean Precision, Mean Recall, Specificity, and Mean Error Rates are utilized. These metrics primarily focus on gauging the model's ability to accurately differentiate between toxic and nontoxic content. Although crucial for identifying the most accurate model, this assessment is insufficient, especially in the context of real-time information networks such as social media platforms.

In selecting optimal toxicity classifiers for social media platforms, accuracy, F1-score, ROC-AUC, etc., are not the only critical factors. Other elements, such as throughput, play a pivotal role. The notion of throughput becomes increasingly significant due to the need for fast processing of vast amounts of data within seconds in the dynamic realm of social media networks. Refer to Section 4.2 for a more comprehensive exploration of these metrics.

Therefore, this study represents a pioneering effort that not only prioritizes the creation of high-accuracy DL models but also places significance on their fast processing capabilities, managing thousands of data points per second. The research introduces an innovative simulation framework, which assists social media companies in the strategic choice of efficient classifiers across diverse toxicity levels. This framework factors in operational costs, advertising revenue, user satisfaction, and visibility of advertisements. Recognizing the influence of toxicity detectors on user satisfaction and revenue, the study underscores the need to consider more than just accuracy when selecting a toxicity classifier. User satisfaction and engagement emerge as crucial elements that impact revenue generation.

3 Methodology

This section delves into the intricacies of our PDS framework, providing insights into how it was developed. Initially, we highlight common features shared by most social media platforms in Google AdSenseGSMM. Following that, we detail the simulation process in Section 3.3, explaining how we simulated various social media components and assessed the impact of different classifiers on users and overall profit.

To illustrate the structure and features of the PDS framework, we begin by offering an overview of the input graph used in the simulation, along with how user behavior is defined and controlled throughout the simulation (explained in Section 3.3.1). Subsequently, in Section 3.3.2, we provide a comprehensive explanation of how the simulation operates, covering the calculation of costs, revenue, and profit.

Finally, in Section 3.3.3, we subject the simulation to null cases to validate its accuracy before employing it for experimentation.

3.1 Generic Social Media Model (GSMM)

We will briefly cover some of the common features shared by most social media platforms (see Figure 1). The primary feature is the **user** who can create and share information on the platform. After creating an account, users can set up a profile, and they are then considered part of the platform's user base. Users can choose to leave the platform at any time and return later. In addition, they have the option to follow or unfollow other users, with each user having a list of followers and followings. The list of followers shows users who

follow the user, while the following list displays users that the user follows.

A **post**, also known as user-generated content, can be in the form of text (unstructured), visual, or audio. Users can view posts generated by other users in their following list and interact with them by reacting or replying. Social media platforms offer various ways for users to react, such as commenting to express opinions, offer praise, disagree, ask questions, and participate in online conversations about social content. Credibility is another significant feature of social media, where users can indicate their agreement or disagreement with a post using features such as "Likes" on Facebook or "Retweets" on Twitter, which also facilitate the dissemination of information.

On a page called the **front page**, users can find a list of the most recent posts from their following list, which are available for a specified period. Meanwhile, the **trending page** showcases the most popular posts determined by a generic algorithm that takes into account factors such as the time since the post was made and the number of comments or likes it has received.

The maintenance of social media platforms greatly relies on **advertisement** revenue as their primary funding source [69]. For example, Google discloses an annual advertising revenue of $50.57 billion, which represents 91% of their yearly earnings. Furthermore, according to a report by Statista [1], advertising constituted 98.3% of Facebook's revenue and 84% of Twitter's revenue in the first quarter of 2021.

Taking these general features into account, we simulate the social media environment. More details on the development and simulation process are described in Section 3.3. It is important to note that, while we utilize these generic features in our study, they may vary for different platforms.

3.2 Environments

As discussed in Section 2.2, it is evident that social media is not uniformly toxic. Numerous factors play a direct role in influencing users' encounters with or the creation of toxic content.

Initially, the likelihood of users engaging or receiving toxic content hinges on their network affiliation and follower/following ratio. Within the realm of social media, symmetric connections, where users mutually follow each other and often share mutual friends, are commonplace, termed as friendship networks. In this context, the likelihood of encountering toxic content is notably low, although not completely absent. Essentially, toxic interactions are less probable between individuals who share direct connections such as friendships, family ties, work associations, and more. However, such instances can still arise due to differing viewpoints, competition, and aspirations for status.

Another essential aspect to consider is the reach rate, which measures the number of users who come across content from an account. Some users receive many comments, reactions, or reposts, which brings their posts to the attention of a wide audience [24]. For instance, this is often the case with celebrities, influencers, and public figures, commonly referred to as "high-reach" users in this paper, who have substantial reaching rates from both followers and non-followers. Whether these users encounter

[1] https://www.statista.com/statistics/279456/share-of-revenue-of-facebook-by-source/

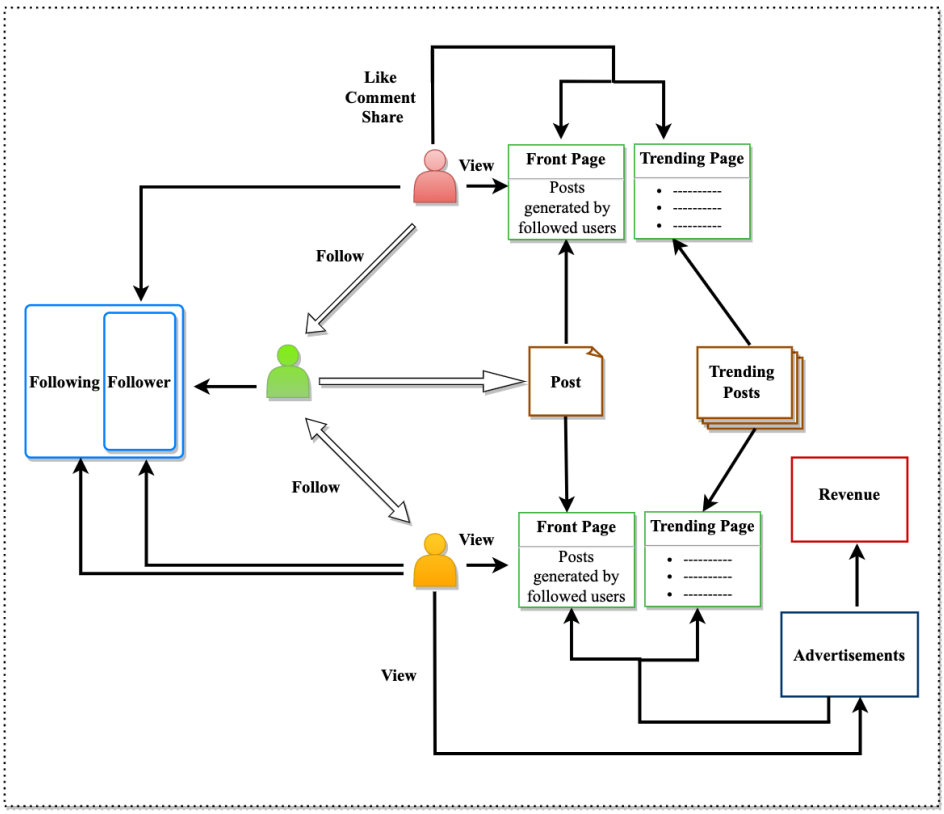

Fig. 1 The architecture of social media

hurtful comments depends on their expertise and their audience. Influencers, activists, and politicians, for example, are more likely to face such negativity. Also, users who post a lot, especially those who get into arguments, are more likely to be attacked or criticized. As the reach rates increase, these users become more vulnerable to encountering toxic content, even if they themselves generate less toxic content. It is exceptionally uncommon for high-reach users to respond negatively to comments from their followers

Moreover, for high-reach users, the balance between the number of their followers and the limited number of accounts they follow is another crucial aspect in addition to the reaching rate. These users often have a significantly larger number of followers compared to the few accounts they follow. This discrepancy increases the chances of encountering toxic comments. In a broader context, interactions that involve users who lack any previous connection, indicating that they do not follow each other, tend to be more toxic. The concept of reaching rates is also applicable to the content they share. High-reach users' posts, along with public content and trending pages, typically reach a larger audience compared to personal posts from users with fewer followers or private accounts. As a result, both creators of widely seen posts and users who frequently comment face an increased risk of encountering toxic

content. In summary, environments characterized by frequent highly visible posts often exhibit a higher frequency of toxic occurrences.

Hence, the varied nature of toxicity within social media underscores the necessity for an appropriate toxic content detector that effectively aligns with this diversity. A single detector cannot reliably perform optimally in different environments. Additionally, the implementation of classifiers customized for each specific environment has significant potential for profitability for social media companies. Subsequent paragraphs will delve into this hypothesis with more comprehensive details.

Consequently, we define three levels of social media toxicity: **low**, **medium**, and **high**, determined by factors such as the user network and the reach rate. Within these toxicity tiers, we will evaluate the effectiveness of the detectors in diverse settings.

3.3 Social Media Profit Simulation

3.3.1 Input Graph and User Behavior

Our consideration involves a group of users connected to each other, with their own list of followers and followings. Users can create posts visible to their followers, and interact with posts by reading, liking, and commenting. We focus only on users who do not want to be exposed to toxic content, although some users may view social media as a platform for self-expression and do not mind encountering toxic content.

User engagement on online platforms can affect the likelihood of encountering harmful content, and personal attacks may lead to decreased activity or platform abandonment [17]. Inaccurate toxicity detection systems can result in users receiving multiple toxic messages (false negatives) and leaving the platform to avoid further exposure.

Disengagement from a platform can occur due to various reasons beyond user sensitivity to harmful content. For instance, frequent post-flagging as toxic can lead to disengagement, particularly when false positives are produced by the classifier. While accurate blocking (true positive) may not significantly affect user engagement, the effect of false positives can be significant. Classifier performance typically has a significant impact on user engagement, with a well-functioning classifier potentially increasing engagement. Simulation frameworks can be used to compare the impact of different toxicity classifiers under various conditions and evaluation metrics. The PDS framework assesses user behavior according to three factors that can cause dissatisfaction with their social media experience (see Figure 2). It is important to note that there is a pre-established limit for the number of posts received for different reasons, and exceeding this limit may lead to user disengagement and platform abandonment. The reasons are as follows:

Toxic posts: The simulation tracks the number of visible toxic posts (false negative) received by users. If a user is exposed to too many toxic posts, they may become disinterested in the platform and leave, resulting in zero revenue for the company. User engagement is measured by the amount of time they spend reading content each day, and a decrease in reading time may be a result of the platform's toxic content.

Blocked posts: The simulation records the number of blocked posts (true

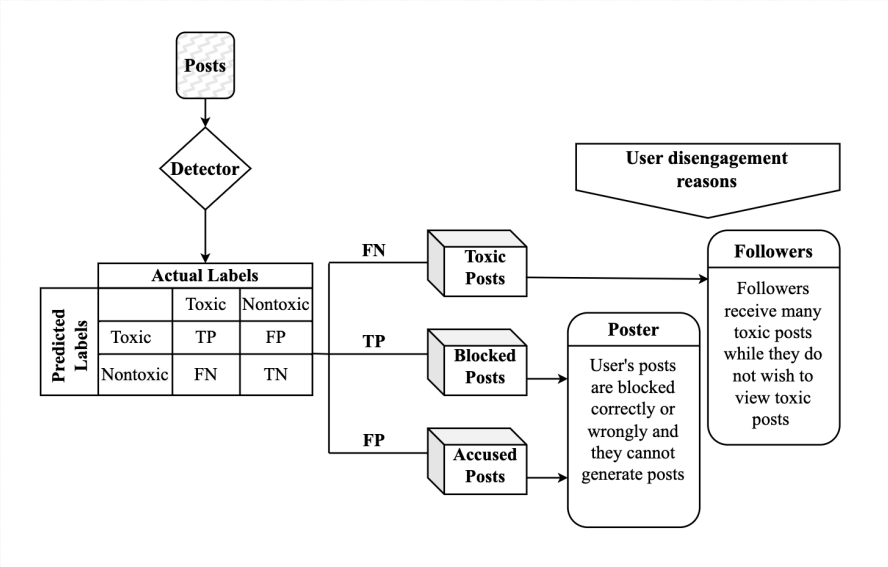

Fig. 2 Different user disengagement reasons

positive) for each user due to the toxicity detector's evaluation. If the detector is too stringent in its assessment, it may block several posts that are not highly toxic, or if it is biased in its classification of toxic content.

Accused Posts: The simulation records and keeps track of the number of false positives for each user, where nontoxic posts are wrongly detected as toxic.

3.3.2 How the PDS Framework works

The simulation system incorporates multiple inputs, such as the user graph, pre-trained classifiers, and previously calculated predictions. As mentioned earlier, the user graph comprises users, their followers, followings, and different sensitivity levels.

In order to simulate the real-time system used by social media, the PDS framework involves the concurrent operation of three distinct threads, as illustrated in Figure 3, detailed below.

Reader Thread: The simulation involves the replication of user activities such as reading, commenting, and liking posts. To accomplish this, we have defined three distinct states namely **Waiting**, **Read Session**, and **Read Post**. Initially, all users are in the waiting state. They are subsequently assigned to their unique reading session using a designated function. Each reading session contains all the new posts available on the front page and the trending page, which are specific to each user. The front page displays all posts created by the user's followings, while the trending page features popular posts whose creators are not on the reader's followings list. Notably, the trending page is dynamic and changes at the end of the simulated day. In order to guarantee that all posts are available and not concealed, snapshots of the front and trending pages are captured when a user

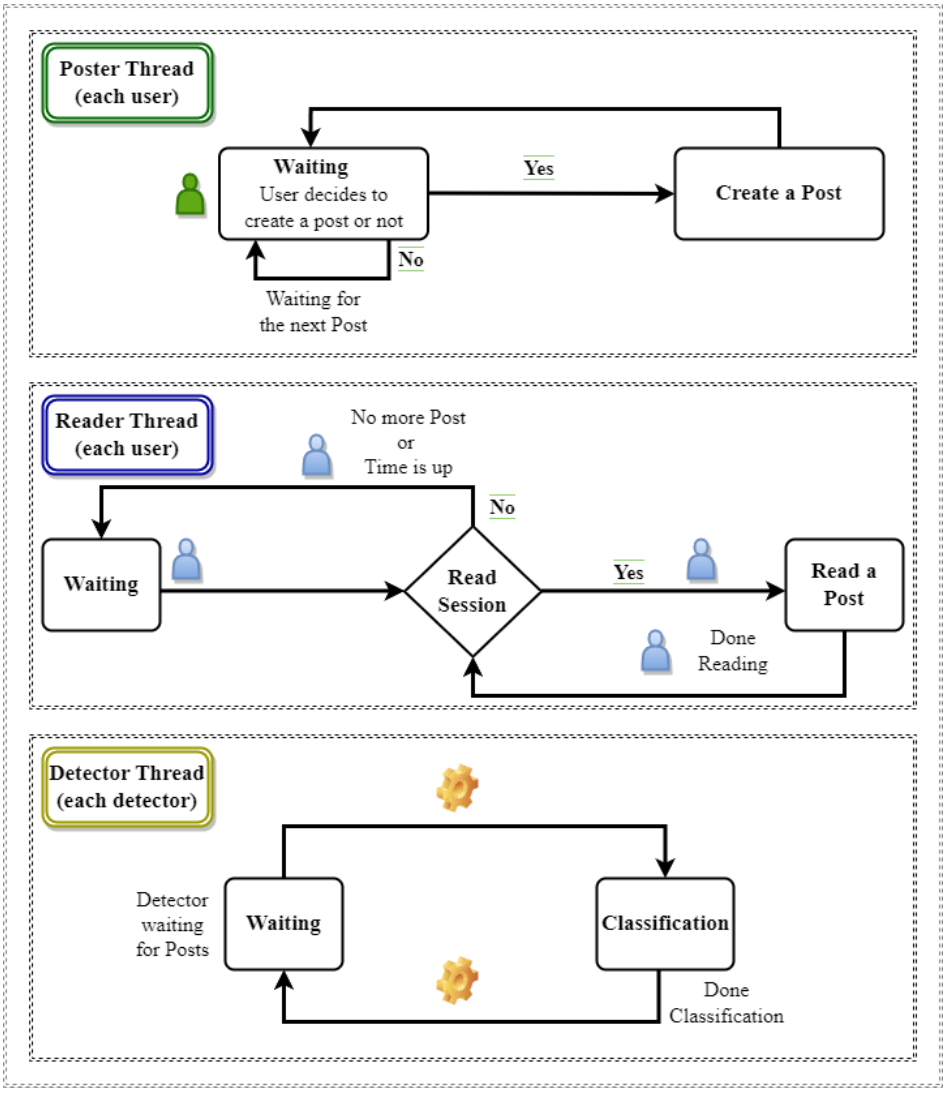

Fig. 3 Architecture for the PDS Framework

enters the reading session. These snapshots are later inspected to identify any posts that have been removed. Once a toxicity detector is implemented to distinguish between toxic and nontoxic posts, some posts may be hidden from the user based on the detector's classification.

We made the assumption that users would first read all unread posts from the people they follow before moving on to unread posts from the trending page, if time permits. During the Reading session, the time spent by users is recorded. Reading sessions are scheduled for a certain amount of time, and some users may read in many small sessions throughout the day, while others may read in fewer or larger chunks. If a user reads a post,

they are moved to the next state, Reading post, and then return to the reading session to read the next post. The time spent in the Reading post state is also recorded. Users are permitted to follow or unfollow the creators of posts they are reading. If a reader decides to follow the creator of an additional post while reading it, their list of followings will expand during the simulation. As a result, the number of posts that they can read will increase, potentially leading to a decrease in the number of readers leaving their reading session early. Users can also like and/or comment on each post during the reading session. A user's engagement increases when they like or comment on a post, while reading toxic content decreases their engagement. When the allotted time for the reading session is over, the user returns to the waiting state until they decide to check for updates and view generated content again.

For the creation of the trending page, we take into account the number of likes (α) and the number of comments (β) received by each public post, along with the time it takes (γ) to accumulate those likes and comments. This information is used to compute the post's score using the formula:

$$\textbf{Score} = \alpha w_1 + \beta w_2 - \gamma w_3 \qquad (1)$$

where $w_1, w_2, w_3 > 0$ and $w_1 + w_2 + w_3 = 1$.

By ranking posts (created within a simulated time) in descending order based on their scores, we select the top 100 posts as trending posts for updating the trending page. The trending page is refreshed every simulated day and prior to users entering their reading session, the system checks if an update to the trending page is needed.

Poster Thread: It simulates how users create new content (toxic and nontoxic). It only includes the Waiting state, where all users are present and can decide whether or not to post. A time will be assigned for each user. Based on their unique probability of generating toxic posts provided by the user graph, a Normal distribution (Gaussian distribution) determines whether the user creates a toxic/nontoxic post. If the user decides to generate a post, then a post will be created. Moreover, the system keeps track of the time per user for generating a post, indicating when each post is generated. This means that the time interval between each post generated by a user can be determined. It should be noted that prior to receiving newly generated posts from the Poster Thread, certain posts have already been forwarded to the Reader Thread, providing users with some content to peruse during the simulation.

Detection Thread: which simulates detecting and hiding toxic posts in real-time. This thread includes two states, Waiting and Processing. A detector is first in the waiting state until it receives posts as input for classifying. Note that, all three threads happen simultaneously, then the number of available posts for processing is different every time. Additionally, we considered different batch sizes for detectors to take input for classification, which resulted in different processing times.

In terms of time, the PDS framework runs on a compressed timescale, where we can simulate a long-running simulation within a short span of time, while still operating in a real-time system. In addition, precalculation is done within the simulation to speed up the system so that we can perform long-running simulations.

Social media toxicity is influenced by the user's network and reaching rate. Toxic content is less likely in friendship networks but still possible. Users with high-reaching rates are more likely to receive toxic comments, especially on high-reaching posts. Therefore, we have categorized toxicity into three levels: low, medium, and high, and we will evaluate the detectors' efficacy in different environments across these three levels of toxicity.

Finally, the outputs of the simulation include **profit**, **revenue** from user engagement, **costs** of the machine for using classifiers, the number of users who did not leave the platform, the number of uncreated posts (because the user left the platform), the **confusion matrix** (error matrix) of the detection results, the total number of posts created by users, and the total number of blocked posts because of toxicity. Note that a confusion matrix is used to summarize the performance of classifiers. This method is useful for measuring Precision, Recall, Accuracy, F1-score, and AUC-ROC curves which will be explained later in Section 4.2.

Cost: We precompute the actual time that a detector takes to do the batch computation and then using a linear transform between the two estimates the amount of computation time will be calculated. To calculate the cost of a classification task using instances, we first determine the total cost per second for an instance used to run the batch classification. Then, the total cost for each classifier is calculated by multiplying the cost per second of the instance used for batch classification and the total detection time. In this context, $\mathbf{W}_i$ represents the cost incurred by instance **i** when processing and classifying samples within a given time frame. Similarly, $\mathbf{T}_{x,i}$ denotes the total time taken by classifier **x** running on instance **i** to complete the classification of **S** samples. Therefore, the total cost for classifying n samples using instance (i) per classifier (x) is given by the following formula:

$$\mathbf{T}_x(\mathbf{t,\ s}) = \sum_{s \in S} t_s \tag{2}$$

where

- $\mathbf{T}_{x,i}\mathbf{(t,\ s)}$ is the total time for processing S samples by classifier x.
- $\mathbf{t}_s$: time required for classifying sample s.

therefore, the total cost is equal to:

$$\mathbf{C(x,\ i)} = T \times W \tag{3}$$

Revenue: As mentioned earlier in Section 3.1, advertising is the primary source of revenue for social media companies. Consider that every post includes an advertisement, which generates revenue per view when users view it. By acknowledging that every post incorporates an advertisement, revenue is generated in accordance with the views it accumulates from users. In other words, each view encompasses both content and ads, and a higher number of views results in more revenue. This is due to the platform receiving payments for showcasing advertisements to its users, effectively leading to increased revenue.

Assume A_v represents the revenue earned per advertisement's view. Let u $\in U_m$ is a user on platform **m**, and $\mathbf{N(u,\ m)}$ shows the total number of views by user **u** on the platform **m**. Then

$$\mathbf{V} = \sum_{u \in U} N(u, m) \tag{4}$$

V represents the total number of views on platform **m** that have been carried out by **U** users. Thus, the total revenue generated from viewing posts is given by:

$$\mathbf{R} = A \times V \tag{5}$$

Profit: Profit can be calculated by subtracting the total cost from the revenue generated.

$$\mathbf{P}_{x,i} = R - C \tag{6}$$

$\mathbf{P}_{x,i}$ represents the profit generated by classifier x, while the **R** and **C** determine the revenue and cost associated with the application of classifier **x** using instance **i**, respectively.

In this study, we used the simplest form of profit. Since the simulation tracks user departures caused by toxic exposure, wrongful blocking, and false accusations, this formulation also partially reflects long-term retention, as users who leave no longer generate revenue. This profit definition is not fixed and can be extended with additional factors such as brand reputation, regulatory compliance costs, legal liability, or demographic-based advertising pricing, depending on the priorities and data available to each platform.

3.3.3 Null Case Verification

The main goal of this study is to identify efficient techniques for addressing online toxicity and increasing user engagement. Thus, using the PDS framework we will test different scenarios to find the best solution per case. To ensure that the PDS framework meets expectations, we first tested several null cases. All null classifiers have zero computation costs, and all machines are the same type. The key parameters for simulating null cases are presented in Table 1.

- **Perfect Detector (PD):** Always returns the correct detection for toxic or nontoxic.
- **Always Toxic Detector (AT):** Returns toxic for all detections.
- **Always Nontoxic Detector (AN):** Returns nontoxic for all detections.
- **Random Detector (RD):** Equally likely to return toxic or nontoxic for all detections.

The Perfect Detector (PD) is the most ideal detector. It is unlikely that a user will leave this platform because of toxic or accused posts. Nevertheless, blocking posts may cause some users to leave the platform. It is important to note that revenue is derived from views of posts, but not from hidden posts. Therefore, a perfect detector does not generate more revenue since toxic posts will be blocked and no one will see them.

The Always Toxic Detector (AT) is the worst detector. Due to the fact that no post will be visible to users, it will generate zero profit in any environment. Users will eventually leave the platform.

Always nontoxic Detector (AN) does not hide any toxic posts, then users who are sensitive to toxic posts will leave the platform. It generates the most revenue because all posts even toxic ones will be shown to users. The revenue decreases over time due to users leaving the platform because of toxic posts. The worst case happens in a highly-toxic environment, and the highest revenue will be generated in a slightly-toxic environment.

Random Detector (RD) fails to function accurately. The posts are incorrectly classified as toxic or nontoxic in this case. It is possible that some users will leave the platform because of posts that are toxic but the classifier failed to detect

them; or because of posts that are falsely detected and blocked as toxic; or due to blocked posts. The revenue generated by RD varies depending on the environment.

The simulation was run with a shrink factor of 100 ×, enabling it to model a week of activity over the course of 6048 seconds, while incorporating 10,000 users during the process. The use of a shrink factor in the simulation is a common technique used to speed up the execution time of simulations. By reducing the scale of the simulation, it is possible to run it more quickly while still maintaining the overall accuracy of the results. In this case, the shrink factor of 100 allowed the PDS framework to execute much more quickly than it would have if it had run at full scale, while still representing a full week of activity. The four detectors were evaluated across environments with varying levels of toxicity: from low toxicity, indicating approximately 20%, to medium toxicity encompassing around 50%, and highly toxic environments comprising over 80%. In this case, zero cost is assumed, resulting in the profit being equal to the revenue.

As discussed above, revenue is also generated based on the number of views by users. To determine the revenue earned per view of an advertisement, *Google AdSense* [2] was utilized. Selecting the *North America* region and the *People and Society* content category would yield a revenue of $0.0086 per view, according to Google AdSense. This projection suggests that 50,000 monthly views would generate an annual revenue of $5,202. Therefore, all three threads (Poster, Reader, Detector) will begin simultaneously. This results in some posts that are not analyzed by the detector and are displayed to users as is. Table 2 shows the profit generated by each detector in different environments.

Table 1 PDS Framework Parameters for Null Cases Verification

Parameter	Value
Shrink Factor	100 ×
Simulation Duration	604800 sec
Users	10,000
Ad_revenue	$0.0086
Cost	0
Low Toxic ratio	∼20%
Medium Toxic ratio	∼50%
High Toxic ratio	∼80%
Machine Type	Paperspace P4000

The simulation results validate our null hypotheses across multiple environments and confirm the reliability of the simulation. Apparently, PD offers the best performance and profitability regardless of the environment. AT, on the other hand, made no profit because all posts were treated as toxic content and were hidden from users, so they couldn't be seen or read by other users. In all environments, AN outperforms RD.

Figure 4 provides an overview of user fall over the simulation time. It is evident that across all environments, the PD classifier demonstrates the lowest user churn rate when compared to other classifiers. Specifically, PD exhibits the least user attrition in the low-toxic environment, while experiencing the higher user churn in medium- and high-toxic environments. Moreover, AT and PD experience a sudden rise in user fall at the beginning of the simulation, which then gradually increases. Similarly, RD and AN show a notable number of users leaving, with a major churn of more than 7500 users taking place within the initial 2000 seconds, followed by a gradual increase.

According to Figure 5 which shows the number of users who left the platform

[2] https://adsense.google.com/

Table 2 Profit Variation Across Different Environments in Null Cases

Environment	Detector	Profit ($)
Low Toxic	PD	381,990
	RD	40,313
	AT	0
	AN	83,853
Medium Toxic	PD	214,315
	RD	13,568
	AT	0
	AN	23,044
High Toxic	PD	146,164
	RD	10,209
	AT	0
	AN	15,094

per second for various reasons, when PD is applied in all environments, users leave the platform only because they get many posts blocked because of toxicity, not because they receive toxic content or their posts are incorrectly blocked. AN, however, causes many users to leave even in low-toxic environments due to no control over shared content and toxicity. When AN is used, fewer users leave the low toxic environment compared to medium/high toxic environments.

It was evident that AT blocked all posts, so users eventually left the platform after their posts were blocked incorrectly or correctly and there were no posts for followers to read. Additionally, more users left because of wrongly blocked posts in higher and medium-toxic environments than in a low-toxic environment. As toxicity increases (medium and high-toxic), the number of users who leave due to blocked posts by AT increases, while the number of users who leave due to wrongly blocked posts decreases. RD results in users leaving for all three reasons in all environments. In addition, a higher level of toxicity leads to a larger number of left users who don't wish to see toxic content.

Users were tracked throughout the simulation and both their departure and reasons were recorded. PD clearly outperforms other approaches, and the line chart shows that it gradually increases over simulation time, even if it starts out sharply. Initially, AT loses the most users, but their numbers gradually increase over time. In both RD and AN, the maximum number of left users has finally been reached.

In general, the simulation outcomes illustrate that diverse classifiers' effectiveness and efficiency can be evaluated in a range of settings, and user engagement can be tracked throughout the simulation process.

4 Experimental Setup

This section describes our experiment setup, including the toxic dataset for toxicity detection, the graph dataset for our PDS framework, and the classifiers used in the detection thread. We also explain the improvements made to the classifiers for higher throughput and provide details on the computing resources and tools utilized in our experiments.

4.1 Datasets

4.1.1 Toxic text data set

We utilized a dataset from the Google Jigsaw team [6] to train our detectors. This dataset comprises 159,571 Wikipedia comments that have been human-rated for toxicity across six categories: toxic, severe toxic, insult, obscene, threat, and identity hate. Additionally, the dataset contains comments labeled as nontoxic. For classifier development, the dataset was stratified and split into three

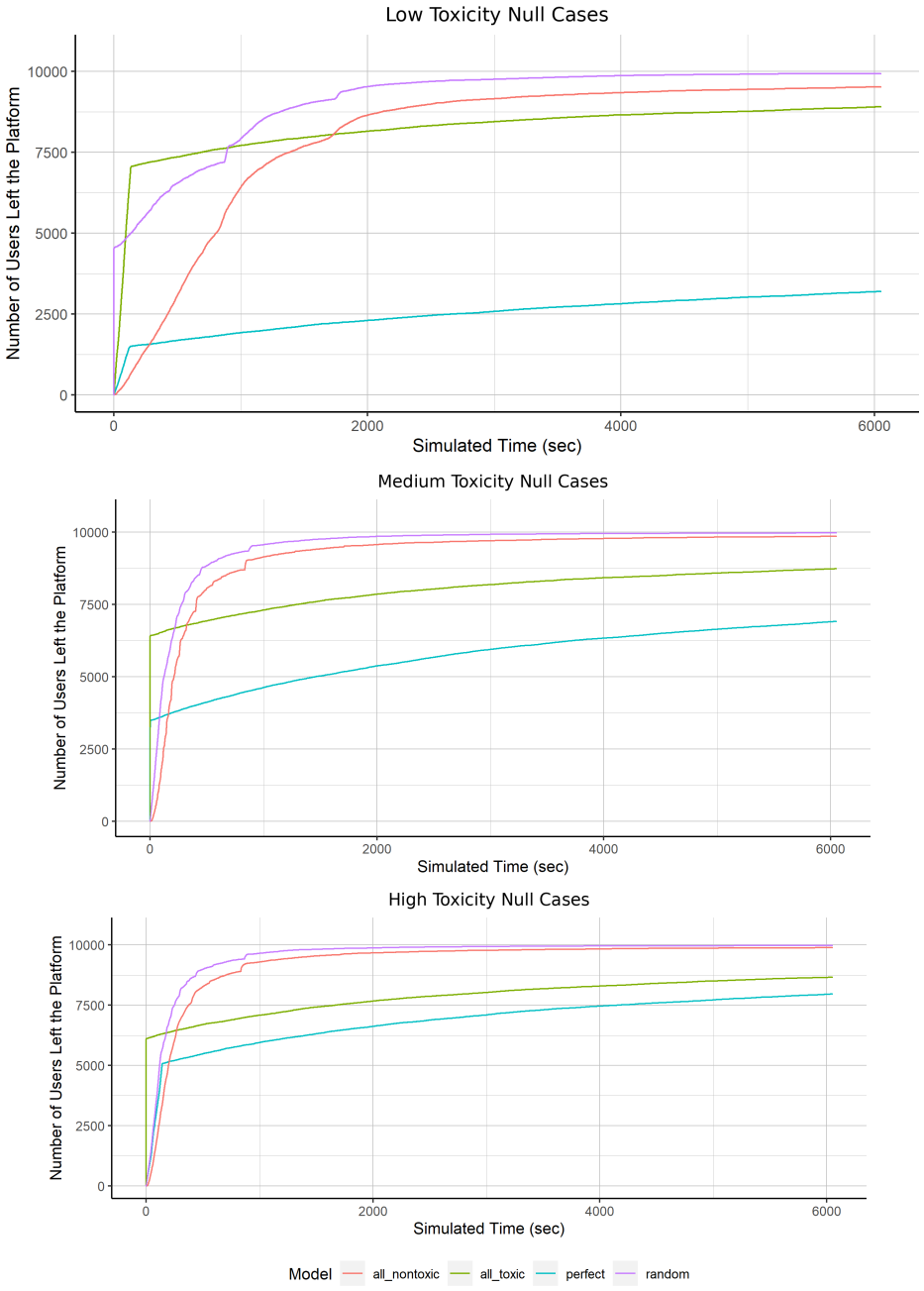

Fig. 4 User churn over the simulation period for Null Cases

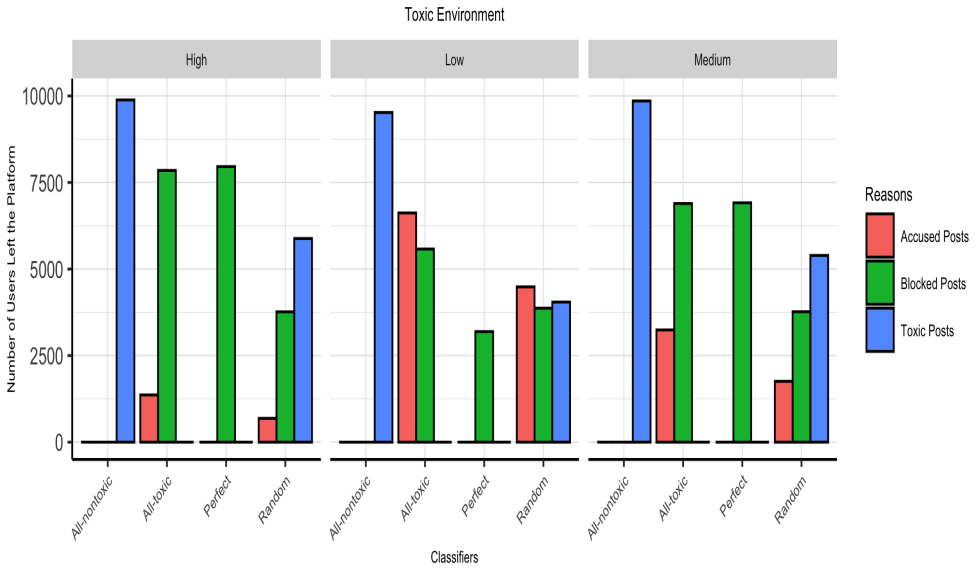

Fig. 5 The number of users left for different reasons based on the classifier used

sets: a training set (70%), a validation set (20%), and a test set (10%).

4.1.2 Graph data set

To better reflect the typical behavior of social media users, a dataset consisting of 10,000 users was generated, with consideration given to various follower/-following ratios. The follower/following ratio, which is calculated by dividing the number of users following an individual (followers) by the number of users they follow (followings), can provide insight into the type of user on social media platforms [70]. Normal users typically have an equal ratio of followers to followings, while celebrities or influencers have a higher number of followers and a lower number of followings, resulting in a high follower/following ratio. New or less active users tend to have a lower number of followers and a higher number of followings. Fake accounts often have few followers, while normal users tend to have identifiable followers such as family and friends. Additionally, profiles can be public or private, allowing users to choose who sees their content. Popular content can be found on the trending page.

To ensure user engagement and retention, each user in the experiment was assigned maximum sensitivity levels based on various factors. For example, a toxic content threshold was assigned to each user using a normal distribution (Gaussian distribution) to represent their sensitivity to toxic content. This is a standard choice for representing heterogeneous individual traits in agent-based and social simulation studies when no established empirical distribution is available in the literature, and it reflects the assumption that most users hold a moderate sensitivity to toxic content, with fewer users at the extremes. Testing alternative distributions as empirical data becomes available is left for future work. Additionally, random numbers were generated for each

user's threshold to account for scenarios such as posts being blocked correctly or incorrectly. Each user also had a unique probability of liking, commenting, following, unfollowing, and creating new posts.

4.2 Detectors

To detect toxic content in social media streams, we developed multiple deep learning classifiers to operate within the Detection Thread of the Profit-Driven Simulation (PDS) framework. These classifiers were selected based on their balance between accuracy and throughput, key factors for real-time deployment. The models include CNN, CNN-FastText, BERT, RoBERTa, and miniature BERT variants such as BERT-Tiny and BERT-Mini. This range offers diversity in architecture and operational trade-offs. While recent large language models (LLMs) like GPT or LLaMA provide high accuracy, they are impractical for high-throughput environments due to their computational cost. Our focus is thus on deployable, efficient models suitable for real-world moderation.

All classifiers were trained and evaluated on the same toxicity dataset to ensure consistency and fairness in comparison.

We evaluated classifier performance using four metrics: Accuracy, AUC-ROC, F1-score, and Throughput. Accuracy measures the proportion of correct predictions. AUC-ROC reflects the model's ability to distinguish between toxic and non-toxic content. F1-score balances precision and recall, offering a single measure of effectiveness. Throughput quantifies how many samples are processed per second, capturing real-time performance.

$$\text{Throughput} = \frac{N}{t} \tag{7}$$

where N is the number of test samples and t is the total inference time.

In total, eight classifier variants were developed, four CNN-based and four Transformer-based. Each group includes a high-accuracy and a high-throughput variant. CNN variants were optimized for speed, while Transformer variants include distilled and miniature BERT/RoBERTa models to improve efficiency. These classifiers are assessed within the PDS framework to determine their effectiveness and cost-efficiency for real-time toxicity detection. Further architectural details and performance enhancements are provided in Sections 4.2.1–4.2.2.

4.2.1 Detectors selected for best accuracy

CNN-based

The input to the CNN model undergoes data cleaning and preprocessing, which involves removing URLs, punctuations, digits, stop-words, and normalizing cases, acronyms, and abbreviations. Tokenization is performed, and the text comments are truncated or padded to a fixed length before being converted to vector words. The model comprises four 1D convolutional layers with multiple window sizes and output channels. A pooling layer is utilized to reduce the dimensionality of the convolutions, followed by a global max pooling to further reduce the dimension. The hidden layers of the model employ ReLU activation function, while the output layer uses Sigmoid activation function with a classification threshold of 0.5. During training, binary cross entropy is used as the loss function to measure the difference between predicted and actual labels. Additionally, Adam optimizer is

employed to update the model's parameters during training, which helps to speed up convergence and improve accuracy.

We incorporated pre-trained fastText word embeddings into our CNN model to accelerate the training process (CNN-fastText). Our experiments employed the "crawl-300d-2M" embeddings, which consist of 2 million word vectors trained on Common Crawl using a continuous bag of words (CBOW) with position weights, character n-grams of length 5, and a window of size 5 and 10 negatives. The CNN model convolves over the embedded vectors, which are then passed through a max pooling layer to reduce the dimensionality. The final classification is performed using fully connected layers with a Sigmoid activation function. The use of pre-trained word embeddings helped to significantly reduce the training time of our CNN model. The hyperparameter tuning for these models was performed using a random search.

The experimental results for the most accurate CNN and CNN-fastText models are shown in Table 3.

According to the results, the CNN model with 30 epochs, 128 batches, 2.00E-05 learning rate, and 0.1 dropout rate achieved the highest accuracy (92%), and F1-score (72.33%) followed closely by the CNN model with fastText embedding, which had a higher AUC, F1-score and throughput. Please note that, while these models are accurate, their throughputs are currently inadequate for effective application in social media scenarios. Hence, we are actively exploring ways to enhance their throughput without compromising their accuracy.

Transformer-based

We utilized the Huggingface transformer library, which is compatible with Tensorflow 2.4.1. To detect toxicity, we tested various versions of BERT such as BERT-base, ALBERT, Distilbert, RoBERTa-base, RoBERTa-large, and DistilRoBERTa. To prepare text data for BERT-based models, we used BERT tokenizer, which transforms raw text into BERT-compliant tokens. A pre-trained BERT model has a built-in tokenizer and can handle up to 512 tokens, which is the maximum limit in BERT. During the training phase, we limited the token length to 200, 256, and 400 tokens to address computational constraints or minimize the risk of overfitting on lengthy sequences. A classification threshold of 0.5 was applied to the output layer for all transformer-based models.

The BERT architecture utilizes the Self Attention mechanism of the Transformer model to acquire context by examining word embeddings in other sentences. We trained our models using different epochs and batch sizes, employed the binary cross-entropy loss function, and Adam adaptive optimizer with varying learning rates for better GPU utilization.

To identify the most accurate BERT and RoBERTa architectures, we conducted experiments and tested various variants by adjusting random hyper-parameters, such as dropout probabilities and ReLU activation. We randomly explored a different set of hyper-parameters for each model to identify the one that produced the most accurate results. We tested the Accuracy, AUC, F1-score, and Throughput values for each model, and their respective optimal hyper-parameters are reported in

Table 3 Classification Experiment Results and Parameters for CNN-based Models with the Best Accuracy

Model	Epoch	Batch size	Learning rate	Dropout rate	Filters	Accuracy	AUC	F1-score	Throughput (s)
CNN	30	128	2.00E-05	0.1	128	0.92	0.70	0.72	2319
CNN fastText Crawl	3	256	0.002	0.4	300	0.92	0.77	0.75	3849

Table 4 for both BERT and RoBERTa models.

The results demonstrate that all BERT and RoBERTa-based models exhibit nearly identical accuracy and F1-score in detecting online toxicity, with variations in throughput. BERT-base has the highest AUC (79%) and F1-score (78%), but the lowest throughput (25). Distilbert has a lower AUC and F1-score compared to BERT-base but a higher throughput (64). RoBERTa-base and DistilRoBERTa are very close in terms of accuracy, AUC, and F-score, but they have different throughputs. However, in the context of social media, where thousands of posts and comments are received every second, faster predictions are necessary. Despite their acceptable accuracy, AUC, and F1-score, transformers have low throughput and require improvements for practical use.

In the upcoming sections, we will detail the modifications applied to the CNN, CNN-fastText, BERT-based, and RoBERTa-based models with the aim of improving their throughput and profitability.

4.2.2 Modified detectors selected for best profit

Although the developed models are accurate in classification tasks, they are not optimal for high-throughput toxicity classification. To address this challenge, a variety of approaches have been suggested and put into practice to find the right balance between throughput and accuracy. However, it's important to note that the classifier with the highest throughput may not necessarily be the optimal choice for toxicity detection and application on platforms. There are numerous other factors that can not be adequately evaluated using metrics such as accuracy or throughput. Therefore, at the end of the assessment, models exhibiting both high throughput and high accuracy will be selected for testing within the PDS framework. Ultimately, the most profitable model, which is best suited for each environment, will be chosen.

CNN-based

(1) Tuning Parameters: A potential avenue for enhancing throughput involves adjusting parameters including learning rates and filters. We experimented with various parameter sets for CNN-based models to evaluate the impact on throughput.

(2) Shallower Network Topology: To potentially increase the throughput of a CNN, we experimented with reducing the depth of the network. While deeper networks can improve accuracy, they can also be computationally expensive and slow down the process. A shallower network with fewer convolutional 1D layers can be a good compromise between accuracy and throughput. For instance, we tested a CNN with four convolution layers and found that while it performed accurately enough, its throughput needed to

Table 4 Classification experiment results and parameters for BERT-based models with the best accuracy

Model	Epoch	Batch size	Learning rate	Dropout rate	Accuracy	AUC	F1-score	Throughput (s)
BERT	3	8	2.00E-05	0.1	0.92	0.79	0.78	25
Distilbert	1	2	2.00E-05	0.2	0.92	0.71	0.75	64
ALBERT	1	16	2.00E-05	0.3	0.92	0.70	0.73	44
RoBERTa	4	16	2.00E-05	0.1	0.92	0.78	0.77	54
DistilRoBERTa	3	16	2.00E-05	0.2	0.92	0.77	0.77	91

be improved. To achieve this, we experimented with reducing the number of convolution layers to three and two, while still maintaining good accuracy.

(3) Smallest pre-trained embedding: Using larger pre-trained models can improve accuracy, but we also tested how smaller pre-trained fastText embedding would affect CNN performance. We compared two fastText pre-trained embeddings: wiki-news-300d-1M (16B tokens) and crawl-300d-2M (600B tokens). The wiki-news-300d-1M, which is trained on Wikipedia 2017, UMBC web-based corpus, and statmt.org news dataset has 1 million word vectors compared to crawl-300d-2M.

It is worth noting that while tuning model parameters and adjusting network topology can improve throughput, hardware and software optimization, such as using specialized hardware and efficient libraries, can also impact it significantly. However, these factors were not considered in this study and will be explored in future work.

After hyperparameter tuning and adjusting the network topology, the CNN model achieved the highest throughput of 12550 per second with 2 convolution layers, batch size of 128, learning rate of 0.01, maximum features of 50000, and filter size of 64. However, it is important to note that only parameter tuning could not significantly improve throughput; a shallower network topology had a more substantial impact. Combining both a shallower network and tuned parameters led to improved throughput. Similar techniques were applied to CNN-fastText, considering different word embeddings. Results revealed that the CNN model without pre-trained embeddings performed best, while among the fastText pre-trained embeddings, the Crawl news model achieved higher throughput than the wiki-news model. The CNN model using Crawl News achieved a throughput of 9364 per second with 2 convolution 1D layers, GlobalMaxPooling1D, batch size of 128, learning rate of 0.01, maximum features of 40000, and filter size of 32. In contrast, the CNN model using wiki-news reached a throughput of 6863 per second with 4 convolution layers. Additional details are available in Table 5 and Table 6 for CNN and CNN-fastText models, respectively.

Transformer-based

Transformers are known for their high computational requirements due to a large number of parameters and multiple layers of transformers. This limitation often makes them unsuitable for real-time applications that require high throughput. To address this issue, we explored various BERT models, including BERT-base, DistilBERT, ALBERT, RoBERTa, and DistilRoBERTa, to identify the most efficient models that offer

Table 5 Throughput Improvement for CNN Models

Throughput (s)	Accuracy	AUC	F1-score	Hidden layers	Filter	Batch Size	Learning Rate
1,694	91.14%	64.50%	61.41%	4	128	512	0.01
2,319	92.06%	70.0%	72.33%	4	128	128	2.00E-05
3,359	90.91%	75.25%	70.59%	4	400	128	5.00E-05
4,845	91.82%	69.98%	70.76%	4	128	256	0.01
6,085	90.98%	73.34%	69.54%	3	300	256	0.002
7,887	90.41%	74.12%	68.73%	3	128	128	0.01
8,651	90.27%	74.0%	68.03%	3	128	64	0.01
10,777	91.40%	73.91%	71.72%	2	128	64	0.01
11,356	91.26%	69.80%	71.37%	2	200	64	0.02
11,934	91.35%	71.38%	68.61%	2	300	64	0.0002
12,550	91.47%	72.12%	70.98%	2	300	64	0.0002

Table 6 Throughput Improvement for CNN-fastText Models

Throughput (s)	Accuracy	AUC	F1-score	Hidden layers	Sources	Filter	Batch Size	Learning Rate
1,860	91.74%	76.83%	73.81%	4	crawl	300	512	0.002
2,028	90.60%	77.25%	74.05%	4	wiki	300	256	0.0002
2,416	91.32%	79.25%	74.26%	4	wiki	300	128	0.002
3,849	92.14%	77.65%	75.25%	4	crawl	300	256	0.002
4,085	91.08%	80.30%	74.03%	4	crawl	300	64	5.00E-5
5,498	91.20%	80.70%	74.96%	3	crawl	300	64	5.00E-5
5,642	91.28%	80.65%	74.52%	3	crawl	300	64	5.00E-5
6,863	90.88%	84.40%	74.71%	2	wiki	128	128	0.001
6,877	91.92%	71.43%	73.28%	4	crawl	128	256	0.01
7,867	91.76%	82.28%	76.39%	2	crawl	64	128	0.001
8,432	91.39%	80.91%	75.04%	2	crawl	64	128	0.0001
9,364	91.74%	73.80%	73.62%	2	crawl	32	128	0.01

high throughput. In addition, we tested all 24 BERT miniatures and found that BERT-Tiny, with two transformer layers and a hidden embedding size of 128, was the most suitable model for our memory and throughput requirements (for more information about BERT miniatures, refer to **Appendix A** (Table 9). Through multiple experiments involving adjustments to parameters, we observed that BERT-Tiny performed optimally with epoch=1, batch-size=16, learning-rate=2.00E-05, and dropout-rate=0.3, yielding an accuracy of 91%, a throughput of 1527 along with an AUC and F1-score of 68.79% and 70.36%, respectively. Among RoBERTa-based variants, DistilRoBERTa achieved the highest throughput at 115 per second, accompanied by accuracy, AUC, and F1-score of 89.76%, 60.03%, and 67%, respectively. A comparison of the throughput among different versions of BERT and RoBERTa is separately illustrated in Figure 6. The results point out that utilizing smaller variants of BERT and RoBERTa embeddings led to higher throughput.

As you observe, classifiers exhibit significant differences in terms of evaluation metrics. The results are derived from testing algorithms in simple experimental setups, which may not accurately

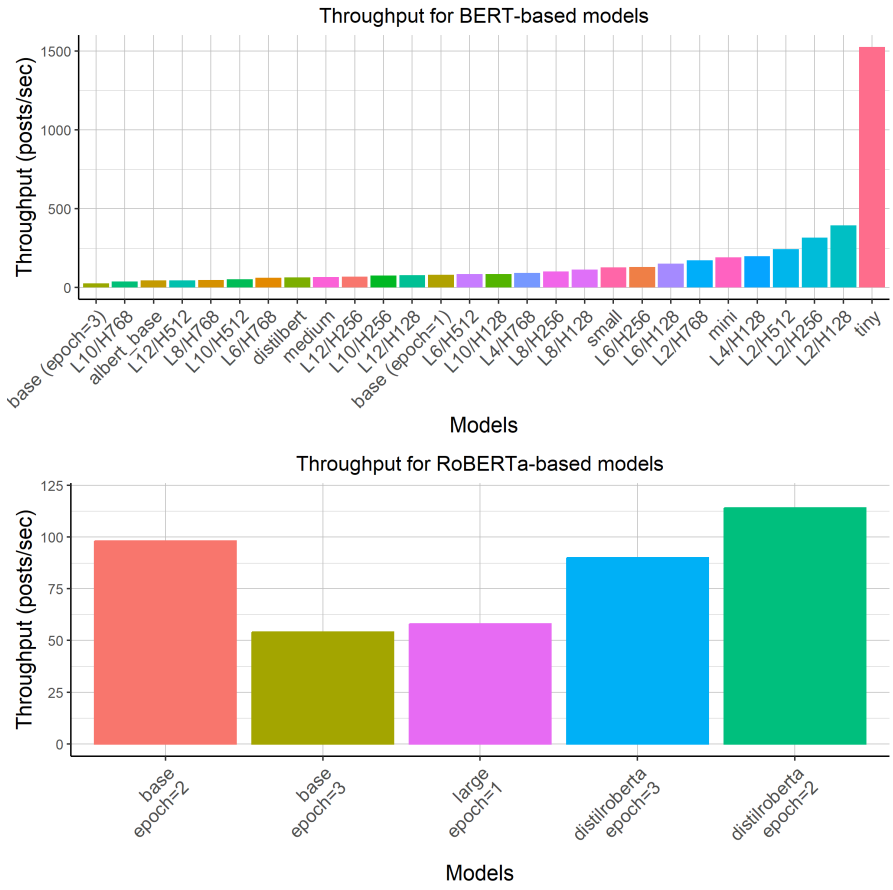

Fig. 6 Throughput for BERT-based and RoBERTa-based Models per second. Note: L = the Number of layers and H = Hidden size.

reflect the complexity of large-scale social media environments. Consequently, we have selected the best models based on their accuracy, F1-score, and throughput to compare and evaluate them using the PDS framework, while other aspects such as cost, revenue, and user satisfaction will also be evaluated this time. Table 7 presents the list of chosen classifiers. Notably, BERT-base boasts the highest accuracy (92.96%) and F1-score (78.26%) among all the models selected for PDS framework, while CNN exhibits the highest throughput (12,550). To differentiate between the models selected for PDS framework, we utilized the suffixes v1 and v2, where v1 indicates the version with higher accuracy, and v2 denotes the version with superior throughput.

4.3 Environments

The evaluation of classifiers spanned across environments exhibiting diverse levels of toxicity. These ranged from low

Table 7 All models selected for experimentation

	Model	Accuracy	AUC	F1-score	Throughput (s)
Accuracy	CNN_{v_1}	92.06%	70.06%	72.33%	2319
	$\text{CNN-fastText}_{v_1}$	92.14%	77.65%	75.25%	3,849
	BERT-base_{v_1}	92.96%	78.50%	78.26%	25
	$\text{RoBERTa-base}_{v_1}$	92.88%	78.63%	77.44%	54
Throughput	CNN_{v_2}	91.47%	72.12%	70.98%	12,550
	$\text{CNN-fastText}_{v_2}$	91.74%	77.65%	75.25%	9,364
	BERT-Tiny_{v_2}	91.50%	68.79%	70.36%	1,527
	$\text{DistilRoBERTa-base}_{v_2}$	89.76%	60.03%	60.67%	115

toxicity, encompassing roughly 20%, to medium toxicity, which included about 50%, and highly toxic environments, constituting over 80%.

4.4 Computer Power

We used the cloud computing instance "Paperspace P4000" with NVIDIA GPUs[3] for our study. P4000 has 1 GPU with 8 GB memory, NVIDIA QUADRO P4000 type, 30 GB RAM, 8 vCPUs, Intel Xeon processor, and 2.60 GHz Clock Speed. The cost per second for a classification task on P4000 has been previously calculated as \$0.000142. Therefore, according to Equation 3, the total cost of each classifier can be determined as follows: $C(x, P4000) = \$0.000142$ multiplied by $T_{x,i}$, where $T_{x,i}$ indicates the total detection time for classifier x running on instance i. Although we wrote all of the classifiers in Python 3.8, we opted to use Java 18 for the PDS framework as it significantly improved the running time and resulted in faster performance. We also visualized the PDS framework results using "InfluxDB"[4], a high-performance, open-source, distributed, and scalable time-series database designed to handle high volumes of time-stamped data. InfluxDB is widely used for real-time monitoring and provides a comprehensive set of tools for data visualization and analysis, making it easy to create real-time dashboards and reports.

4.5 Profit

The ultimate metric employed to evaluate the performance of distinct classifiers is profit. The computation of profit within the PDS framework is carried out using Equation 6. The calculation of costs was elucidated in the previous section. In terms of revenue assessment, the same configuration and features used for verifying null cases were adopted. This process is comprehensively explained in Section 3.3.2. As a quick reminder, the revenue earned per advertisement view is determined by "Google AdSense" at a rate of \$0.0086 per user view for the "North America region". More specifically, within the "People and Society" content category. The total revenue is calculated using Equation 5.

5 Experimental Results

Individually, every classifier was employed as the toxicity detector within the simulation's detector thread to exhibit its real-time performance within scaled environments. These environments

[3] https://www.nvidia.com/en-us/design-visualization/quadro/
[4] https://www.influxdata.com/

encompassed scenarios with 10,000 users over the span of one week, incorporating varying degrees of toxicity. The outcomes are condensed in Table 8 outlining the profit achieved by each classifier in distinct environmental conditions.

The results show that in a low-toxic environment, the model with the highest throughput is the most profitable solution. A CNN_{v_2} with 91.47% accuracy, 70.98% F1-score and 12550 throughput produces $402,046 profit while a CNN_{v_1} with 92.06% accuracy, and 72.33% produces $343,759 profit. Although BERT_{v_1} is the most accurate model, it could not be the most profitable in this situation. Moreover, the profits produced by BERT_{v_1} ($341,727) are very close to those produced by CNN_{v_1} ($343,759), while their throughputs are very different. Even though $\text{DistilRoBERTa}_{v_2}$ has a higher throughput (115) than other RoBERTa variants, it achieves the lowest profit ($88,796) among all models due to its low accuracy (89.76%).

Under moderately toxic conditions, CNN_{v_1} generated the highest profit of $70,807 outperforming other models. There are other classifiers, such as CNN-fastTextv_1, that exhibit higher accuracy, AUC, F1-score, and throughput when compared to CNNv_1. However, despite their superior performance, they were unable to generate higher profits. It is interesting to note that neither the model with the highest throughput nor the most accurate model made the most profit in this environment. Similar to the low-toxic environment, $\text{DistilRoBERTa}_{v_2}$ makes the lowest profit. In the medium-toxicity environment, the mix of toxic and nontoxic content is roughly even, creating a regime where classification accuracy has a larger marginal effect on user retention. As shown in Figure 8, the dominant reason for user departure in this environment shifts toward toxic posts reaching users, rather than blocked posts. CNN_{v_1}, with its higher accuracy and F1-score, catches more toxic content before users are exposed to it, retaining users longer and generating more revenue despite its lower throughput. In contrast, in low- and high-toxicity environments, throughput plays a more dominant role: in low toxicity there is less toxic content to catch, while in high toxicity the volume is so large that even accurate classifiers cannot prevent most users from eventually leaving. The reasoning makes sense, given that when a classifier underperforms in a low-toxic environment, while having neither higher accuracy nor throughput than others, it will likely underperform in a medium or high-toxic environment.

As a result of a highly toxic environment, CNN_{v_2} with the highest throughput ranks first in terms of profit ($26,196). Next, it is followed by the most accurate model (BERT-base_{v_1}) which generates $24,143 profits. With a profit of $22,201, CNN_{v_1} takes third place. In line with our expectations, $\text{DistilRoBERTa}_{v_2}$ has the worst performance with a profit of $15,162.

According to Figure 7, 36.05% of users left low-toxic environments when CNN_{v_2} was applied, while 95.4% left by $\text{DistilRoBERTa}_{v_2}$. The total percentage of users left over the simulation varied between 50% and 80% for other classifiers. There is a sharp increase of 58% in $\text{DistilRoBERTa}_{v_2}$, followed by a gradual increase, while in other classifiers the rate gradually increases from the beginning. All classifiers encountered the maximum number of left users in medium and high-toxic environments. Compared to high-toxic environments, medium-toxic environments have a slower rate of user fall.

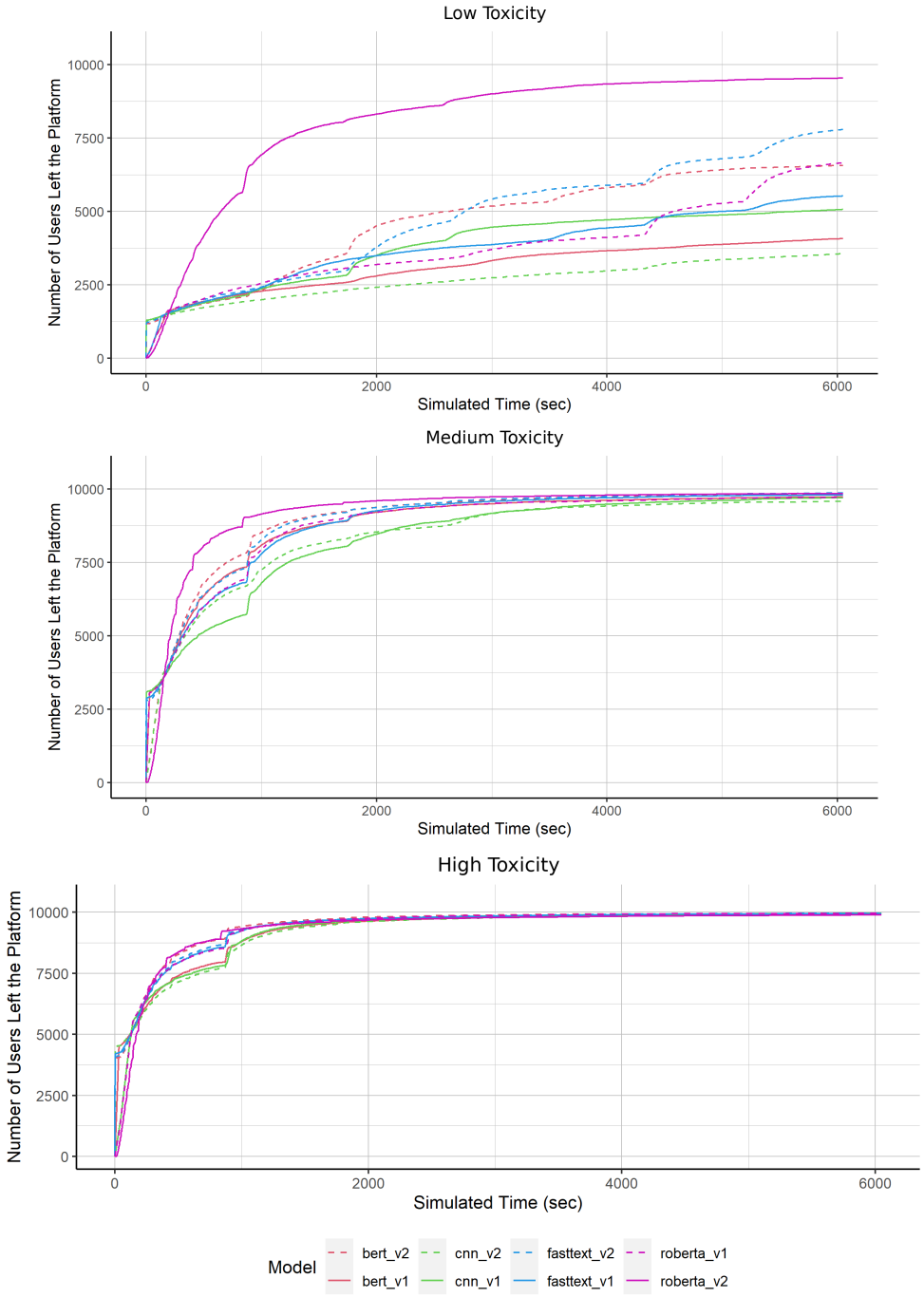

Fig. 7 Users churn over the simulation time

There are fewer left users in CNN_{v_1} until 22000 seconds, but CNN_{v_2} meets it and shows fewer falls in the rest of the test. CNN_{v_2} keeps increasing very closely to CNN_{v_1} (90.55%) in a highly toxic environment, but CNN_{v_2} finally achieves the lowest number (99.3%).

In addition, the number of users who left the platform for different reasons is also shown by Figure 8. The results demonstrate that by applying CNNv_2 in low- and high-toxic environments, the main reason that pushes users away from the platform is a large number of blocked posts, while in a medium-toxic environment, the main reason for leaving the platform is a large number of toxic posts. Moreover, applying RoBERTav_2 in all environments resulted in user churn due to a large number of toxic posts.

Overall, the CNN_{v_2} classifier outperformed the others in low and highly toxic environments, but CNN_{v_1} was the most profitable classifier in the medium-toxic environment. The study shows that CNN-based models were the best candidates under the tested assumptions and parameter settings for detecting toxic language online.

6 Conclusion, Limitations and Future Work

This study presented the Profit-Driven Simulation (PDS) framework for evaluating toxicity detection models under realistic, real-time social media conditions. Unlike traditional offline benchmarks, PDS incorporates operational constraints such as throughput, computational cost, latency, and user engagement, enabling deployment-oriented model assessment.

Experiments with eight classifiers showed that no single model performs optimally across all scenarios. High-throughput models are better suited to low- and high-toxicity environments, while moderately accurate and efficient models perform more effectively in medium-toxicity settings. These results emphasize the importance of context-aware model selection beyond static accuracy metrics.

This study has several limitations. The simulation results are based on a single set of parameter settings, economic assumptions, and hardware configuration (Paperspace P4000); relative rankings between classifiers may shift under different parameter values or on different hardware, as architectures such as CNNs and Transformers can benefit unevenly from specialized inference infrastructure. The profit model also uses a simplified formulation, and no real-world deployment or external behavioral validation has been performed. These limitations should be considered when interpreting the results.

Future work includes large-scale validation using real-world social media data, expanding the diversity of evaluated models and datasets, and improving the efficiency of Transformer-based approaches through optimization techniques. The PDS framework can accommodate any classifier in the detection thread, so evaluating large language models such as GPT and LLaMA is a natural direction for future work as efficient inference techniques for these models continue to develop. Repeating simulation runs across multiple independent seeds to report confidence intervals and statistical comparisons, particularly where profit differences between classifiers are small, and validating the framework's conclusions across different hardware configurations, are also important next steps.

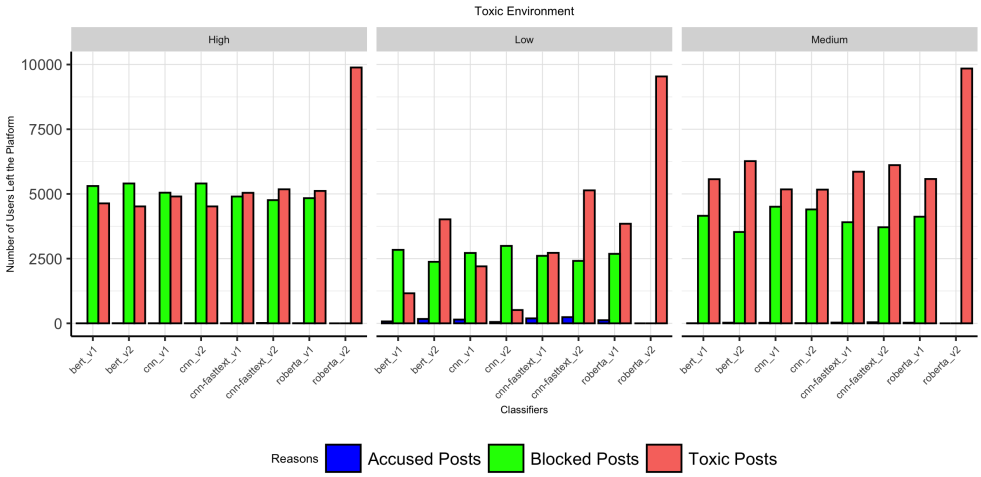

Fig. 8 The reasons for leaving a platform in different environments by applying detectors

Table 8 PDS framework results for different classifiers

Model	Profit ($)		
	Low	Medium	High
CNN_{v_1}	343,759	**70,807**	22,201
$CNN\_fastText_{v_1}$	311,540	47,186	19,243
$BERT\_base_{v_1}$	341,727	46,678	24,143
$RoBERTa_{v_1}$	312,819	48,000	18,667
CNN_{v_2}	**402,046**	63,123	**26,196**
$CNN\_fastText_{v_2}$	282,555	39,945	18,371
$BERT\_Tiny_{v_2}$	279,594	37,016	16,783
$DistilRoBERTa_{v_2}$	88,796	23,017	15,162

Finally, integrating fairness considerations and assessing long-term impacts on user trust remain essential for responsible and sustainable deployment.

Acknowledgements. The authors gratefully acknowledge Scrawlr (https://scrawlr.com/) for providing support for this research. We also thank Kexin Yan, Software Engineer at Scrawlr, for technical assistance and valuable discussions during the development of the simulation framework.

Declarations

- Funding: The research is supported by MITACS through the Accelerate and Business Strategy programs. In addition, the research is also supported in part by Discovery Grants (RGPIN-2024-04087) from the Natural Sciences and Engineering Research Council of Canada (NSERC), Canada Research Chairs Program (CRC-2019-00041), and R&D Spearhead Grants

Table 9 Characteristics of 24 Smaller BERT Models (English only, Uncased, Trained with WordPiece masking

	H=128	**H=256**	**H=512**	**H=768**
L=2	2/128 (BERT-Tiny)	2/256	2/512	2/768
L=4	4/128	4/256 (BERT-Mini)	4/512 (BERT-Small)	4/768
L=6	6/128	6/256	6/512	6/768
L=8	8/128	10/256	10/512	10/768
L=10	10/128	10/256	10/512	10/768
L=12	12/128	12/256	12/512	12/768 (BERT-Base)

(2024-1301) from the National Cybersecurity Consortium (NCC).

- Conflict of interest: The authors declare no conflict of interest.
- Data availability: The data set generated and analyzed during the current study is not publicly available.
- Authors' contributions: All authors reviewed the manuscript.

Appendix A

Table 9 illustrates detailed information about the BERT miniatures and their characteristics.

References

[1] Vogels, E.a.: The State of Online Harassment (2021). https://www.pewresearch.org/internet/2021/01/13/the-state-of-online-harassment/ Accessed 2022-03-30

[2] Kocoń, J., Figas, A., Gruza, M., Puchalska, D., Kajdanowicz, T., Kazienko, P.: Offensive, aggressive, and hate speech analysis: From data-centric to human-centered approach. Information Processing & Management **58**(5), 102643 (2021) https://doi.org/10.1016/j.ipm.2021.102643 . Accessed 2023-10-30

[3] Ghenai, A., Noorian, Z., Moradisani, H., Abadeh, P., Erentzen, C., Zarrinkalam, F.: Exploring hate speech dynamics: The emotional, linguistic, and thematic impact on social media users. Information Processing & Management **62**(3), 104079 (2025) https://doi.org/10.1016/j.ipm.2025.104079 . Accessed 2025-04-12

[4] Ravikumar, N., Kumaresan, P.K., Rajiakodi, S., CN, S., Chakravarthi, B.R.: Nalvakku: fact-based counter narratives for Tamil hate speech. International Journal of Data Science and Analytics **22**(1), 57 (2026) https://doi.org/10.1007/s41060-026-01023-x . Accessed 2026-08-18

[5] Maslej-Krešňáková, V., Sarnovský, M., Butka, P., Machová, K.: Comparison of Deep Learning Models and Various Text Pre-Processing Techniques for the Toxic Comments Classification. Applied Sciences **10**(23), 8631 (2020) https://doi.org/10.3390/app10238631 . Number: 23. Accessed 2021-10-02

[6] Jigsaw, G.: Jigsaw Toxic Comment Classification Challenge (2017). https://kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge Accessed 2022-04-26

[7] Yan, R., Li, Y., Li, D., Wang, Y., Zhu, Y., Wu, W.: A Stochastic

Algorithm Based on Reverse Sampling Technique to Fight Against the Cyberbullying. ACM Transactions on Knowledge Discovery from Data **15**(4), 1–22 (2021) https://doi.org/10.1145/3441455 . Accessed 2023-08-25

[8] Beniwal, R., Maurya, A.: Toxic Comment Classification Using Hybrid Deep Learning Model. (2021). https://doi.org/10.1007/978-981-15-8677-4_38

[9] Hern, A.: Social network giants pledge to tackle abuse of women online. The Guardian (2021). Chap. Society

[10] Jo Dixon, S.: Biggest social media platforms by users 2025. Publication Title: Statista (2025). https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/ Accessed 2025-04-12

[11] Wu, X.-K., Zhao, T.-F., Lu, L., Chen, W.-N.: Predicting the Hate: A GSTM Model based on COVID-19 Hate Speech Datasets. Information Processing & Management **59**(4), 102998 (2022) https://doi.org/10.1016/j.ipm.2022.102998 . Accessed 2025-04-12

[12] Jo Dixon, S.: Social media - statistics & facts. Publication Title: Statista (2024). https://www.statista.com/topics/1164/social-networks/?utm_source=chatgpt.com Accessed 2025-04-12

[13] Patel, J.: An Effective and Scalable Data Modeling for Enterprise Big Data Platform. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 2691–2697. IEEE, Los Angeles, CA, USA (2019). https://doi.org/10.1109/BigData47090.2019.9005614

[14] Fortuna, P., Nunes, S.: A Survey on Automatic Detection of Hate Speech in Text. ACM Computing Surveys **51**(4), 85–18530 (2018) https://doi.org/10.1145/3232676 . Accessed 2023-03-29

[15] Sheth, A., Shalin, V.L., Kursuncu, U.: Defining and detecting toxicity on social media: context and knowledge are key. Neurocomputing **490**, 312–318 (2022) https://doi.org/10.1016/j.neucom.2021.11.095 . Accessed 2026-08-18

[16] Kirkpatrick, D.: Google: 53% of mobile users abandon sites that take over 3 seconds to load. Publication Title: Marketing Dive (2016)

[17] Urbaniak, R., Ptaszyński, M., Tempska, P., Leliwa, G., Brochocki, M., Wroczyński, M.: Personal attacks decrease user activity in social networking platforms. Computers in Human Behavior **126**, 106972 (2022) https://doi.org/10.1016/j.chb.2021.106972 . Accessed 2022-03-30

[18] Contardo, G.: Machine learning under budget constraints. phdthesis, Université Pierre et Marie Curie - Paris VI (July 2017). https://tel.archives-ouvertes.fr/tel-01677223 Accessed 2022-04-27

[19] Li, W.: A Content-Based Approach for Analysing Cyberbullying on Sina Weibo. In: Proceedings of the 2019 2nd International Conference on

Information Management and Management Sciences. IMMS 2019, pp. 33–37. Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3357292.3357294

[20] Chang, J.P., Cheng, J., Danescu-Niculescu-Mizil, C.: Don't Let Me Be Misunderstood:Comparing Intentions and Perceptions in Online Discussions. In: Proceedings of The Web Conference 2020, pp. 2066–2077. Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3366423.3380273 Accessed 2022-04-29

[21] Santos, C.N.d., Melnyk, I., Padhi, I.: Fighting Offensive Language on Social Media with Unsupervised Text Style Transfer. arXiv. arXiv:1805.07685 [cs] (2018). http://arxiv.org/abs/1805.07685 Accessed 2022-09-20

[22] Wadud, M.A.H., Kabir, M.M., Mridha, M.F., Ali, M.A., Hamid, M.A., Monowar, M.M.: How can we manage Offensive Text in Social Media - A Text Classification Approach using LSTM-BOOST. International Journal of Information Management Data Insights **2**(2), 100095 (2022) https://doi.org/10.1016/j.jjimei.2022.100095 . Accessed 2022-09-20

[23] Wang, W., Chen, L., Thirunarayan, K., Sheth, A.P.: Cursing in English on twitter. In: Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing, pp. 415–425. ACM, Baltimore Maryland USA (2014). https://doi.org/10.1145/2531602.2531734

[24] De Salve, A., Mori, P., Guidi, B., Ricci, L., Pietro, R.D.: Predicting Influential Users in Online Social Network Groups. ACM Transactions on Knowledge Discovery from Data **15**(3), 1–50 (2021) https://doi.org/10.1145/3441447 . Accessed 2023-08-25

[25] Weltevrede, E., Helmond, A., Gerlitz, C.: The Politics of Real-time: A Device Perspective on Social Media Platforms and Search Engines. Theory, Culture & Society **31**(6), 125–150 (2014) https://doi.org/10.1177/0263276414537318 . Accessed 2023-08-18

[26] Gehl, R.W.: The archive and the processor: The internal logic of Web 2.0. New Media & Society **13**(8), 1228–1244 (2011) https://doi.org/10.1177/1461444811401735 . Accessed 2023-08-18

[27] Founta, A.-M., Djouvas, C., Chatzakou, D., Leontiadis, I., Blackburn, J., Stringhini, G., Vakali, A., Sirivianos, M., Kourtellis, N.: Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior. arXiv (2018). https://doi.org/10.48550/arXiv.1802.00393

[28] Qayyum, H., Zhao, B.Z.H., Wood, I.D., Ikram, M., Kaafar, M.A., Kourtellis, N.: A deep dive into the consistently toxic 1% of Twitter. arXiv. arXiv:2202.07853 [cs] (2022). http://arxiv.org/abs/2202.07853 Accessed 2022-09-22

[29] Jhaver, S., Boylston, C., Yang, D.,

Bruckman, A.: Evaluating the Effectiveness of Deplatforming as a Moderation Strategy on Twitter. Proceedings of the ACM on Human-Computer Interaction **5**(CSCW2), 381–138130 (2021) https://doi.org/10.1145/3479525 . Accessed 2022-09-22

[30] Saveski, M., Roy, B., Roy, D.: The Structure of Toxic Conversations on Twitter. In: Proceedings of the Web Conference 2021. WWW '21, pp. 1086–1097. Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3442381.3449861

[31] Radfar, B., Shivaram, K., Culotta, A.: Characterizing Variation in Toxic Language by Social Context. Proceedings of the International AAAI Conference on Web and Social Media **14**, 959–963 (2020). Accessed 2022-04-28

[32] Jahan, M.S., Oussalah, M.: A systematic review of Hate Speech automatic detection using Natural Language Processing. arXiv:2106.00742 [cs] (2021). arXiv: 2106.00742. Accessed 2022-04-27

[33] Fortuna, P., Soler-Company, J., Wanner, L.: How well do hate speech, toxicity, abusive and offensive language classification models generalize across datasets? Information Processing & Management **58**(3), 102524 (2021) https://doi.org/10.1016/j.ipm.2021.102524 . Accessed 2026-03-03

[34] Pamungkas, E.W., Basile, V., Patti, V.: Towards multidomain and multilingual abusive language detection: a survey. Personal and Ubiquitous Computing **27**(1), 17–43 (2023) https://doi.org/10.1007/s00779-021-01609-1 . Accessed 2026-03-03

[35] Hardage, D., Najafirad, P.: Hate and Toxic Speech Detection in the Context of Covid-19 Pandemic using XAI: Ongoing Applied Research (2020). Accessed 2023-08-25

[36] Malik, P., Aggrawal, A., Vishwakarma, D.K.: Toxic Speech Detection using Traditional Machine Learning Models and BERT and fastText Embedding with Deep Neural Networks. In: 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), pp. 1254–1259 (2021). https://doi.org/10.1109/ICCMC51019.2021.9418395

[37] Bodaghi, A., Fung, B.C.M., A. Schmitt, K.: AugmenToxic: Leveraging Reinforcement Learning to Optimize LLM Instruction Fine-Tuning for Data Augmentation to Enhance Toxicity Detection. ACM Trans. Web (Special Issue on Advances in Social Media Technologies and Analysis) (2024) https://doi.org/10.1145/3700791

[38] Wijesiriwardene, T., Inan, H., Kursuncu, U., Gaur, M., Shalin, V.L., Thirunarayan, K., Sheth, A., Arpinar, I.B.: ALONE: A Dataset for Toxic Behavior Among Adolescents on Twitter. In: Aref, S., Bontcheva, K., Braghieri, M., Dignum, F., Giannotti, F., Grisolia, F., Pedreschi, D. (eds.) Social Informatics. Lecture Notes in Computer Science, pp. 427–439. Springer, Cham (2020). https://doi.org/10.1

007/978-3-030-60975-7_31

[39] Mossie, Z., Wang, J.-H.: Vulnerable community identification using hate speech detection on social media. Information Processing & Management **57**(3), 102087 (2020) https://doi.org/10.1016/j.ipm.2019.102087 . Accessed 2023-04-05

[40] Mall, R., Nagpal, M., Salminen, J., Almerekhi, H., Jung, S.-G., Jansen, B.J.: Four Types of Toxic People: Characterizing Online Users' Toxicity over Time. In: Proceedings of the 11th Nordic Conference on Human-Computer Interaction: Shaping Experiences, Shaping Society, pp. 1–11. ACM, Tallinn Estonia (2020). https://doi.org/10.1145/3419249.3420142

[41] Androcec, D.: Machine learning methods for toxic comment classification: a systematic review. Acta Universitatis Sapientiae, Informatica **12**, 205–216 (2020) https://doi.org/10.2478/ausi-2020-0012

[42] Rahul, Kajla, H., Hooda, J., Saini, G.: Classification of Online Toxic Comments Using Machine Learning Algorithms. In: 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 1119–1123 (2020). https://doi.org/10.1109/ICICCS48265.2020.9120939 . https://ieeexplore.ieee.org/document/9120939 Accessed 2026-08-18

[43] Saif, M.A., Medvedev, A.N., Medvedev, M.A., Atanasova, T.: Classification of online toxic comments using the logistic regression and neural networks models. AIP Conference Proceedings **2048**(1), 060011 (2018) https://doi.org/10.1063/1.5082126 . Accessed 2023-08-21

[44] Bonetti, A., Martínez-Sober, M., Torres, J.C., Vega, J.M., Pellerin, S., Vila-Francés, J.: Comparison between Machine Learning and Deep Learning Approaches for the Detection of Toxic Comments on Social Networks. Applied Sciences **13**(10), 6038 (2023) https://doi.org/10.3390/app13106038 . Number: 10. Accessed 2023-08-21

[45] Akuma, S., Lubem, T., Adom, I.T.: Comparing Bag of Words and TF-IDF with different models for hate speech detection from live tweets. International Journal of Information Technology **14**(7), 3629–3635 (2022) https://doi.org/10.1007/s41870-022-01096-4 . Accessed 2023-08-21

[46] HaCohen-Kerner, Y., Miller, D., Yigal, Y.: The influence of preprocessing on text classification using a bag-of-words representation. PLOS ONE **15**(5), 0232525 (2020) https://doi.org/10.1371/journal.pone.0232525 . Accessed 2023-08-21

[47] Rupapara, V., Rustam, F., Shahzad, H.F., Mehmood, A., Ashraf, I., Choi, G.S.: Impact of SMOTE on Imbalanced Text Features for Toxic Comments Classification Using RVVC Model. IEEE Access **9**, 78621–78634 (2021) https://doi.org/10.1109/ACCESS.2021.3083638 . Conference Name: IEEE Access

[48] Zhao, R., Yan, R., Chen, Z., Mao, K., Wang, P., Gao, R.X.: Deep learning and its applications to machine

health monitoring. Mechanical Systems and Signal Processing **115**, 213–237 (2019) https://doi.org/10.1016/j.ymssp.2018.05.050 . Accessed 2022-05-03

[49] Salim, C.E.R., Suhartono, D.: A Systematic Literature Review of Different Machine Learning Methods on Hate Speech Detection. JOIV: International Journal on Informatics Visualization **4**(4), 213–218 (2020) https://doi.org/10.30630/joiv.4.4.476 . Accessed 2022-05-03

[50] Vaidya, A., Mai, F., Ning, Y.: Empirical Analysis of Multi-Task Learning for Reducing Identity Bias in Toxic Comment Detection. Proceedings of the International AAAI Conference on Web and Social Media **14**, 683–693 (2020) https://doi.org/10.1609/icwsm.v14i1.7334 . Accessed 2023-08-21

[51] Wang, Z., Zhang, B.: Toxic Comment Classification Based on Bidirectional Gated Recurrent Unit and Convolutional Neural Network. ACM Transactions on Asian and Low-Resource Language Information Processing **21**(3), 51–15112 (2021) https://doi.org/10.1145/3488366 . Accessed 2023-08-21

[52] Ibrahim, M., Torki, M., El-Makky, N.: Imbalanced Toxic Comments Classification Using Data Augmentation and Deep Learning. In: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 875–878 (2018). https://doi.org/10.1109/ICMLA.2018.00141

[53] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, ??? (2016). http://www.deeplearningbook.org

[54] Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning. ICML '08, pp. 160–167. Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1390156.1390177

[55] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. arXiv (2013). https://doi.org/10.48550/arXiv.1301.3781

[56] Pennington, J., Socher, R., Manning, C.: Glove: Global Vectors for Word Representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543. Association for Computational Linguistics, Doha, Qatar (2014). https://doi.org/10.3115/v1/D14-1162

[57] Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., Joulin, A.: Advances in Pre-Training Distributed Word Representations. arXiv. arXiv:1712.09405 [cs] (2017). http://arxiv.org/abs/1712.09405 Accessed 2022-09-19

[58] Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T.: Learning Word Vectors for 157 Languages. arXiv (2018). https://doi.org/10.48550/arXiv.1802.06893

[59] Bojanowski, P., Grave, E., Joulin,

A., Mikolov, T.: Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017) https://doi.org/10.1162/tacl_a_00051 . Accessed 2023-08-25

[60] MOHAMMED, H.H., DOGDU, E., GÖRÜR, A.K., CHOUPANI, R.: Multi-Label Classification of Text Documents Using Deep Learning. In: 2020 IEEE International Conference on Big Data (Big Data), pp. 4681–4689. IEEE, Atlanta, GA, USA (2020). https://doi.org/10.1109/BigData50022.2020.9378266

[61] Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Technical Report arXiv:1810.04805, arXiv (May 2019). https://doi.org/10.48550/arXiv.1810.04805 . arXiv:1810.04805 [cs] type: article. http://arxiv.org/abs/1810.04805 Accessed 2022-06-15

[62] Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv. arXiv:1802.05365 [cs] (2018). https://doi.org/10.48550/arXiv.1802.05365 . http://arxiv.org/abs/1802.05365 Accessed 2023-02-25

[63] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention Is All You Need. arXiv. arXiv:1706.03762 [cs] (2023). https://doi.org/10.48550/arXiv.1706.03762 . http://arxiv.org/abs/1706.03762 Accessed 2026-01-13

[64] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A Lite BERT for Self-supervised Learning of Language Representations (2019) https://doi.org/10.48550/arXiv.1909.11942 . Accessed 2022-10-31

[65] Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv (2020). https://doi.org/10.48550/arXiv.1910.01108 . http://arxiv.org/abs/1910.01108 Accessed 2026-03-03

[66] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv. arXiv:1907.11692 [cs] (2019). https://doi.org/10.48550/arXiv.1907.11692

[67] Zhao, Z., Zhang, Z., Hopfgartner, F.: A Comparative Study of Using Pre-trained Language Models for Toxic Comment Classification. In: Companion Proceedings of the Web Conference 2021. WWW '21, pp. 500–507. Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3442442.3452313

[68] Fan, H., Du, W., Dahou, A., Ewees, A.A., Yousri, D., Elaziz, M.A., Elsheikh, A.H., Abualigah, L., Al-qaness, M.A.A.: Social Media Toxicity Classification Using Deep Learning: Real-World Application UK Brexit. Electronics **10**(11), 1332 (2021) https://doi.org/10.3390/electronics10111332 . Number: 11. Accessed 2023-08-21

[69] Zafarani, R., Tang, L., Liu, H.: User Identification Across Social Media. ACM Transactions on Knowledge Discovery from Data **10**(2), 1–30 (2015) https://doi.org/10.1145/2747880 . Accessed 2023-08-25

[70] Manikonda, L., Hu, Y., Kambhampati, S.: Analyzing User Activities, Demographics, Social Network Structure and User-Generated Content on Instagram. arXiv. arXiv:1410.8099 [physics] (2014). http://arxiv.org/abs/1410.8099 Accessed 2022-10-11