Browser-Based Detection of Harmful Content with Deep Learning Model

Ni Made Deni Sikiandani [1]*, I Made Agus Dwi Suarjaya [2]*, Yohanes Perdana Putra [3]*
* Teknologi Informasi, Universitas Udayana
** denisikiandani35@gmail.com [1], agussuarjaya@it.unud.ac.id [2], yperdana.putra@gmail.com [3]

Article Info

Article history:

Received 2025-06-13
Revised 2025-07-25
Accepted 2025-07-30

Keyword:

Harmful Content,
Deep Learning,
BiLSTM,
Whisper Automatic Speech Recognition,
Browser Extension

ABSTRACT

This study presents a browser extension that detects harmful content on both web pages and TikTok using a deep learning-based approach. The core model employs a Bidirectional Long Short-Term Memory (BiLSTM) network for multi-label classification, targeting six categories: Toxic, Severe Toxic, Obscene, Threat, Insult, and Identity Hate. The dataset combines 13,057 labeled samples from a public Kaggle dataset (2021) and 2,884 manually labeled tweets scraped from Twitter (X) between October–November 2024. Three feature extraction methods were tested: learned embeddings, FastText, and Word2Vec. The BiLSTM model architecture includes one embedding layer, a 32-unit bidirectional LSTM, three dense layers (128,256,128) using ReLU activation, and a six-unit sigmoid output layer. The model was trained using the Adam optimizer and binary cross-entropy loss, with early stopping applied after five stagnant validation checks across a maximum of 200 epochs. While the FastText-based model showed the best performance, the final deployed model used learned embeddings in Scenario 1 due to its smaller size (1.6M parameters) and near-optimal performance (Recall: 0.9786; Hamming Loss: 0.0052). The extension also integrates Whisper ASR for detecting harmful speech in video-based platforms like TikTok and supports five customizable censorship filters. User evaluation via Customer Satisfaction Score (CSAT) indicated strong acceptance, with 95.45% rating the user experience as Excellent, 84.09% confirming detection relevance, and 79.55% rating the system performance as Good. This highlights the extension's effectiveness in promoting safer digital interaction across text and audiovisual content.

I. INTRODUCTION

In the current digital era, internet usage in Indonesia continues to rise significantly each year. As of early 2024, internet penetration in Indonesia reached 66.5%, with approximately 185.3 million individuals connected out of a total population of 278.6 million [1]. Of these internet users, 139 million use the internet for social media purposes, meaning around 75% of active internet users are engaged with social media platforms [1]. Social media offers many benefits, including ease of communication, information sharing, social networking, and self-expression. However, the freedom of expression on these platforms sometimes leads to negative consequences, such as the rise of hate speech, hoaxes, and other harmful content [2]. These issues have contributed to the proliferation of harmful content that is often inadequately filtered.

Harmful content in this study refers to any form of digital media whether videos or words that may cause offense, distress, or psychological harm to individuals or groups [3]. In textual form, it may include hate speech, cyberbullying, threats, profanity, verbal harassment, slurs, or discriminatory language based on race, religion, gender, or identity. In video-based content, harmful material may appear in the form of spoken hate speech or offensive language embedded in audio tracks. For example, a video might contain verbal abuse directed at a specific ethnic group, sexually explicit language used to harass an individual, or provocative speech that incites violence or spreads misinformation. The presence of such content makes social media environments unsafe and uncomfortable, threatening social cohesion and potentially triggering conflict and division [4].These cases remain prevalent today. For instance, in early November 2024, Indonesian law enforcement handled 32 cases of defamation within just five days [5]. This situation highlights the need for more effective preventive systems to detect and filter harmful content. Although regulations such as Indonesia's Electronic Information and Transactions Law (UU ITE) are in place, many harmful contents go undetected due to the reactive nature of these laws, which rely on user reports and cannot automatically filter harmful content on digital platforms.

A webpage is a single digital page accessible via the internet and identified by a unique Uniform Resource Locator (URL), which may be configured as public or private by the developer [6], [7]. Webpages have become integral to daily digital life, with high traffic across a wide array of websites. These include social media platforms like TikTok, knowledge-sharing forums like Quora, open-source encyclopedias like Wikipedia, and digital news portals all of which serve as primary sources of information, entertainment, and opinion for users. The widespread and growing usage of these platforms signifies a major shift in digital media consumption, driven by content-based interactions.

TikTok exemplifies this trend, with 1.59 billion active users worldwide as of January 2025, representing 27.5% of the global population aged 18 and above [8]. Its popularity, especially among teenagers and young adults, makes it one of the most influential content distribution platforms today. TikTok's short-form, interactive videos allow for rapid content creation, sharing, and engagement. However, like many other digital platforms, this freedom and volume of content also open the door to harmful content such as hate speech, harassment, racism, and provocation.

Similar concerns arise across other platforms, whether in articles on news portals, discussions on Quora, or edits on Wikipedia. While these platforms aim to promote informative and constructive interactions, they are not immune to misuse. Therefore, creating automated detection systems based on deep learning has become increasingly essential for ensuring a safe and healthy digital environment.

The decision to employ deep learning (DL) over traditional machine learning (ML) methodologies for this task is deliberate, stemming from DL's fundamental advantages in processing complex, unstructured data like text and speech. Traditional ML models, often termed shallow machine learning, depend heavily on handcrafted feature engineering, a process where experts must manually define and extract relevant features from the data [9]. This approach is not only time-consuming, labor-intensive, and inflexible but also struggles to capture the intricate contextual nuances inherent in harmful content. In contrast, deep learning overcomes this limitation through its core capability of automated feature learning [9]. Deep neural network architectures can hierarchically learn and discover discriminative features directly from raw data such as words in a text or transcribed audio with minimal human intervention. This ability is crucial, as harmful language is often dynamic and highly contextual. By automatically identifying these complex patterns, deep learning provides a more robust and adaptive approach to effectively detect the multifaceted nature of harmful content, thereby outperforming shallow ML models for tasks involving text and audio data.

Deep learning, a branch of machine learning modeled after the structure of the human brain using Artificial Neural Networks (ANNs), is extensively utilized to address complex tasks like image recognition, natural language processing, and speech recognition [10]. Existing technological solutions include hate speech detection [11], toxic comment detection [12] and abusive language detection [13], which primarily use deep learning models. However, most of these solutions focus solely on text data, making it difficult to detect harmful content in video formats automatically indicating a gap that requires a more comprehensive and effective approach.

Integrating deep learning with the Whisper Automatic Speech Recognition (ASR) model within a browser extension presents a more efficient solution for detecting harmful content in real time as users browse webpages. A promising deep learning model for this purpose is Bidirectional Long Short-Term Memory (BiLSTM) is a type of Recurrent Neural Network (RNN) that excels at capturing contextual information from both past and future sequences by processing data in forward and backward directions.

Previous studies have shown that BiLSTM outperforms LSTM in detecting hate speech on Twitter, achieving an accuracy of 80.25% compared to 78.67% for LSTM [14]. Another study comparing GRU, LSTM, BiLSTM, and multi-dense LSTM models for abusive language detection found that BiLSTM achieved a remarkable 99.9% accuracy with the lowest RMSE of 0.01 [13]. Additionally, a study combining text-based deep learning models with speech recognition revealed that the Whisper ASR and BERT combination achieved better performance (77% accuracy) than wav2vec, which directly processed audio without transcription (71.43% accuracy) [15].

These findings demonstrate that BiLSTM performs well in detecting harmful content, and hybrid models combining Whisper ASR with deep learning NLP methods are more effective in detecting speech-based harmful content than audio-only approaches. Therefore, this research proposes the use of a BiLSTM classifier for harmful text detection and Whisper ASR for transcribing speech to text. Unlike previous studies, this research employs a dataset with six labels: toxic, severely toxic, obscene, insult, threat, and identity-based hate. This broader classification allows detection not only of hate speech but also of sexual harassment, threats, and general toxic language. Moreover, while prior studies mainly focused on model development, this research goes further by implementing the models directly into a browser extension enabling real-time, user-facing applications.

Based on the above background, this study implements a BiLSTM deep learning model integrated with the Whisper Automatic Speech Recognition (ASR) model into a browser extension to automatically detect harmful content in both text and video data. The model is capable of classifying harmful content into six distinct categories: toxic, severe toxic, obscene, insult, threat, and identity-based hate. By covering a wide range of harmful behaviors, the system enhances the granularity and precision of content moderation. The primary objective of this research is to promote a safer and more supportive digital environment while enhancing the effectiveness of efforts to prevent the spread of harmful content online.

II. Proposed Method

The approach used in this study to apply a deep learning model for detecting harmful content on web pages via a browser extension involves multiple stages, as depicted in Figure 1. The research begins with a literature review to identify existing methods, models, and technologies relevant to harmful content detection. This is followed by the data collection phase, which gathers data from various sources, including Kaggle and X (formerly known as Twitter).

Once the data is collected, a preprocessing stage is conducted, which includes the application of three feature extraction techniques which are learned embedding, FastText, and Word2Vec. The processed data is then used to train a BiLSTM model, which is subsequently evaluated to determine the best-performing configuration. The model achieving the highest performance is deployed to an API, which is hosted on Google Cloud Platform using the Cloud Run service. Finally, this deployed API is integrated with a browser extension client to enable real-time detection of harmful content on webpages.

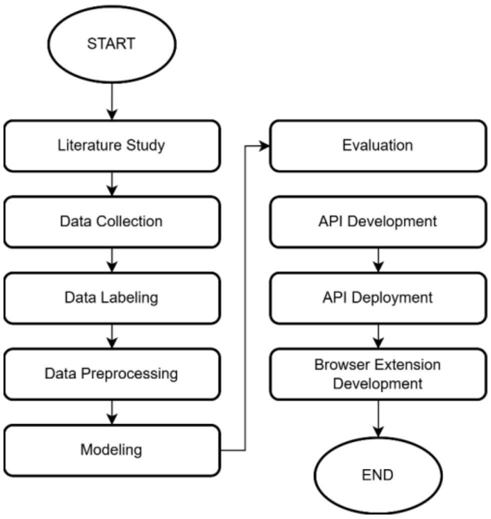


Figure 1. Research Workflow Diagram

A. Data Collection

In this study, the data collection process involved multiple sources to support the development of a deep learning-based browser extension for detecting harmful content. A total of 13,057 rows of labeled text data were obtained from a publicly available Kaggle dataset released in 2021 [16], serving as the primary training data. Additionally, 2,884 tweets were collected from Twitter (formerly known as X) using Tweet Harvest to enrich the dataset with more recent and relevant harmful content examples. These tweets were scraped during the period of October to November 2024.. The collected data subsequently undergoes a labeling process, particularly for the dataset obtained from X, followed by several preprocessing steps aimed at preparing the data for the modeling stage.

B. Data Labeling

The data labeling phase is carried out to obtain classification model data through manual annotation. The data is categorized into six classification labels: toxic, severe toxic, obscene, insult, threat, and identity hate. The descriptions of each class label are presented in Table 1.

TABLE I
Class Label

Class Label	Description
Toxic	Text containing general toxicity.
Severe Toxic	Text exhibiting a high level of toxicity.
Obscene	Text containing obscene content or verbal expressions related to sexual harassment.
Insult	Text containing offensive remarks aimed at an individual or a group.
Threat	Text that conveys threats toward an individual or a group.
Identity Hate	Text expressing hatred toward specific identities, including race, religion, or social groups.

In the dataset, each data row may be associated with more than one label, making this a multi-label classification task. If a data row corresponds to a particular label, the value in the respective label column is set to 1; otherwise, it is set to 0. If a row does not belong to any of the harmful content categories, all label columns will have a value of 0.

C. Data Preprocessing

Text preprocessing is a crucial step in text classification tasks, aiming to convert unstructured raw data into a structured and analyzable format [17], where pre-processing refers to the procedures carried out before entering the processing or data analysis stage [18]. In this study, the preprocessing pipeline consists of several stages. Initially, data cleaning is performed by removing null and duplicate entries to ensure data quality. Case folding is then applied to standardize all characters into lowercase, enabling uniformity in textual representation [17]. Noise removal is conducted to eliminate irrelevant components such as URLs, user mentions, emojis, hashtags, escape characters, numeric values, and repeated characters, which may introduce noise and hinder model performance. This step ensures that the textual data is free from unnecessary symbols that do not contribute meaningful information. Furthermore, text normalization is employed to convert non-standard words into their standard forms. Tokenization is carried out to segment the text into individual tokens by removing delimiters such as spaces, new lines, or tabs [19], which facilitates further processing. Subsequently, stemming is applied to reduce each word to its root form without considering grammatical or semantic context, minimizing vocabulary size and improving generalization. For example, words such as “berlari,” “lari-lari,” and “pelari” are reduced to their root “lari.” Although stemming is effective, it may occasionally yield linguistically incorrect roots due to its reliance on morphological patterns. Finally, stopword removal is conducted to remove frequently used words that have little meaningful contribution, such as “in,” “at,” “the,” and “is” [17]. These preprocessing steps are crucial for improving the quality of the text data and ensuring its effectiveness for subsequent processing. Prior to feature extraction, the input sequences are padded to standardize their lengths, with a maximum of 60 tokens per sequence. This ensures uniform input dimensions for the subsequent feature extraction stage. The study then employs three different feature extraction techniques which are learned embedding, Word2Vec, and FastText to convert the padded text into numerical representations suitable for classification.

1) *Trainable Embedding Layer*: The trainable embedding layer or learned embedding is a supervised learning approach in which word embeddings are initialized with random weights and updated during model training through backpropagation [20]. In this study, the trainable embedding layer provided by the Keras library is utilized, with an embedding matrix dimension of 32.

2) *Word2Vec:* Word2Vec transforms words into dense vectors, positioning those with similar meanings close together in the vector space to reflect their semantic relationships [21]. It comprises two main models: Continuous Bag of Words (CBOW) and Skip-gram. [22]. CBOW predicts a target word by averaging the vectors of surrounding words, offering computational efficiency while overlooking word order. On the other hand, Skip-gram infers nearby context words from a target word, capturing more nuanced semantic relationships but requiring higher computational resources.

3) *FastText:* FastText is a word embedding method that forms part of the feature extraction process by transforming unstructured text into structured features [23]. It is a library developed by Facebook and extends the Word2Vec framework [24]. Unlike Word2Vec, FastText does not treat each word as a whole but instead breaks words into character-level n-grams. For example, the word “pintar” with trigram (n=3) is represented as “pin”, “int”, “nta”, and “tar”. This approach enables FastText to handle out-of-vocabulary words, a limitation in Word2Vec. FastText also supports both CBOW and Skip-gram models, incorporating the n-gram mechanism into their architectures.

TABLE II
FEATURE EXTRACTION CONFIGURATION

Feature Extraction	Dimension	Window Size	n-gram
Leanred Embedding	32	None	None
Word2Vec	400	5	None
FastText	300	5	5

This study employs the Continuous Bag-of-Words (CBOW) model for both Indonesian pre-trained feature extraction methods, Word2Vec and FastText, with model configurations presented in Table 3.

D. Modeling

The deep learning model used in this research is a hybrid architecture that combines the BiLSTM model with the Whisper ASR system.. The BiLSTM model is designed to classify harmful sentences based on their respective labels, while the Whisper ASR is utilized to transcribe audio data into text, enabling the subsequent classification process by the BiLSTM model.

1) *WBiLSTM Model*: The Bidirectional Long Short-Term Memory model is a deep learning architecture derived from Recurrent Neural Networks (RNNs), commonly used for text analysis tasks [12]. BiLSTM consists of two LSTM layers: one reads the input sequence in a forward direction, and the other in reverse. This bidirectional structure allows the model to capture context from both earlier and later parts of the sequence [25]. this dual-state architecture comprising a Forward State and a Backward State enhances the model’s ability to retain sequential dependencies from both directions [26]. Due to this bidirectional nature, BiLSTM is also well-suited for processing time-series data [27]. The architecture diagram of the LSTM model is shown in Figure 2.. The BiLSTM model used in this study begins with an embedding layer, which varies depending on the feature extraction method. For the learned embedding, a vector size of 32 is used. In contrast, the Word2Vec and FastText models utilize pre-trained embeddings with dimensions of 400 and 300. After the embedding layer, the model includes a Bidirectional LSTM layer with 32 units, allowing the network to capture context from both past and future tokens. This is followed by three dense layers with 128, 256, and 128 units using ReLU activation, which help in learning complex patterns. The output layer consists of six sigmoid-activated units corresponding to the multi-label classification targets. The model is trained using the Adam optimizer and binary cross-entropy loss function. Evaluation metrics include binary accuracy, precision, and recall. Training runs for a maximum of 200 epochs with early stopping applied when no improvement is observed after five consecutive validation checks. The selected hyperparameters are based on commonly adopted practices and preliminary experiments. The configuration balances model complexity and training

efficiency, ensuring reliable performance across different types of feature embeddings.

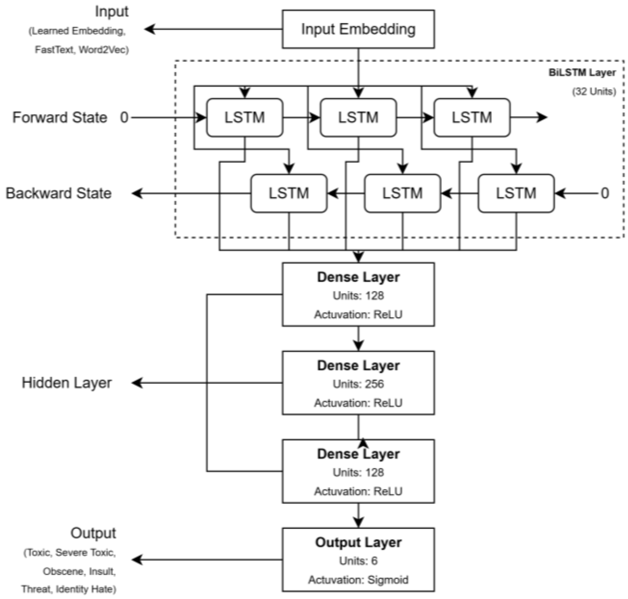


Figure 2. BiLSTM Diagram

1) *Whisper Automatic Speech Recognition Model:* Whisper ASR is a sophisticated speech-to-text system created by OpenAI, intended to transcribe spoken audio into written text. It employs a Transformer-based sequence-to-sequence encoder-decoder architecture [28], which has proven effective for training on large-scale multilingual and diverse audio datasets [29]. Whisper is a ready-to-use model that does not require additional fine-tuning and supports multiple tasks including English transcription, multilingual transcription, translation, and voice activity detection. It preprocesses audio into log-Mel spectrograms, which are encoded using convolutional and Transformer layers. The decoder then generates text from these encoded features using attention mechanisms. Whisper is available in multiple model sizes, each offering trade-offs between performance and computational cost. The word error rate (WER) of the multilingual models consistently declines with an increase in the number of parameters, highlighting the model's scalability and effectiveness across various languages and datasets [29].

E. Evaluation

Evaluation metrics are essential tools used to assess a model's performance, particularly in deep learning and statistical tasks. These metrics evaluate a model's performance on the given data by measuring key aspects such as precision, recall, F1-score, hamming loss, and analyzing the confusion matrix.

In this study, recall is prioritized as the main evaluation metric due to the nature of the task, which involves detecting harmful content. Missing a harmful instance (false negative) can be more critical than incorrectly flagging benign content (false positive). Therefore, maximizing recall is essential to ensure the model captures as many harmful cases as possible, even at the cost of precision. Recall, or sensitivity, assesses how effectively the model identifies all relevant instances belonging to the positive class.. It is defined in equations 1

$$Recall = \frac{TP}{(TP + FN)} \qquad (1)$$

Meanwhile, Hamming Loss is chosen for its suitability in multi-label classification settings, where an instance may be associated with multiple labels to measure the fraction of incorrectly predicted labels per instance [30]. It quantifies the proportion of labels that are misclassified compared to the ground truth, computed separately for each sample. Given a true label vector y and a predicted label vector ŷ for an instance, Hamming Loss is defined as the ratio of misclassified labels. For a dataset with N samples and L possible labels, as shown in equation 2

$$Hamming\ Loss = \frac{1}{nL}\sum_{i=1}^{N}\sum_{j=1}^{L} \vdash\!\dashv [y_{\{i,j\}} \neq \hat{y}_{\{i,j\}}] \qquad (2)$$

Here, the indicator function $\vdash\!\dashv [y_{\{i,j\}} \neq \hat{y}_{\{i,j\}}]$ (XOR) returns 1 if the predicted label differs from the true label (i.e., an error), and 0 otherwise. Hamming Loss is particularly suitable for multi-label problems, where each instance may be associated with multiple correct labels.

Confusion Matrix is a performance summary tool for classification models, especially binary ones. It displays actual and predicted values in a matrix format, showing the counts of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN) which can be seen in Table 4. This matrix helps identify the types of errors the model makes and assess classification performance beyond simple accuracy.

TABLE III
CONFUSION MATRIX

		True Values	
		Positive	Negative
Prediction	Positive	TP	FP
	Negative	FN	TN

F. Application Programming Interface

The API framework used in this study to develop the RESTful API is FastAPI, a Python-based framework. The best-performing model is deployed via the API and integrated with the Whisper model to enable harmful content detection features on TikTok. The workflow of the API integration is illustrated in Figure 2.

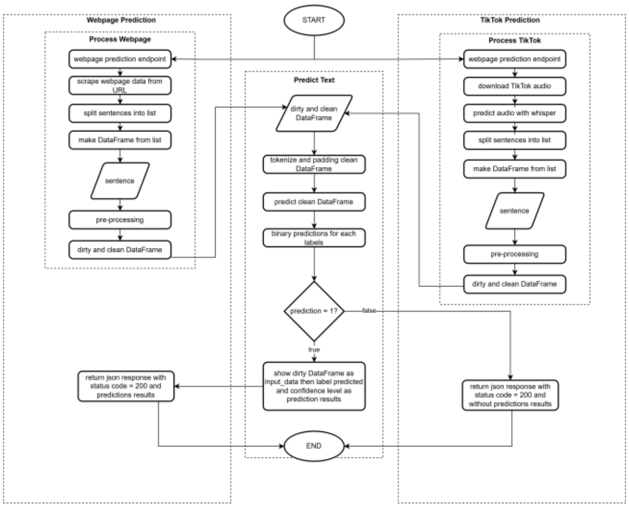

Webpage Prediction
Process Webpage
webpage prediction endpoint
scrape webpage data from URL
split sentences into list
make DataFrame from list
sentence
pre-processing
dirty and clean DataFrame
START
Predict Text
dirty and clean DataFrame
tokenize and padding clean DataFrame
predict clean DataFrame
binary predictions for each labels
prediction = 1?
false
true
show dirty DataFrame as input_data then label predicted and confidence level as prediction results
return json response with status code = 200 and predictions results
END
TikTok Prediction
Process TikTok
webpage prediction endpoint
download TikTok audio
predict audio with whisper
split sentences into list
make DataFrame from list
sentence
pre-processing
dirty and clean DataFrame
return json response with status code = 200 and without predictions results

Figure 2. Workflow of Application Programming Interface

G. *Deployment*

The deployed model on the API locally is further deployed to the cloud using Google Cloud Platform (GCP), specifically utilizing the Cloud Run service. The Cloud Run deployment is configured with 16 GB of RAM, 8 vCPUs, a timeout setting of 180 seconds, and a maximum instance count limited to 1.

H. *Browser Extension*

A browser extension is a lightweight software application that can be installed in a web browser to enhance its functionality or add new features. In this study, the browser extension is developed as a Chrome Extension version 3, utilizing JavaScript as the programming language, while the user interface is built using HTML and CSS. The extension is designed to perform sentence filtering to detect harmful content and display safety notifications for TikTok videos being accessed.

I. *CSAT Score*

The Customer Satisfaction Score (CSAT) is a commonly used metric for gauging user satisfaction with a product, service, or experience, typically gathered via brief Likert-scale surveys (e.g., 1–5 or 1–10). It is calculated by dividing the number of positive ratings (typically scores of 4 or 5) by the total number of responses, then multiplying the result by 100 to obtain a percentage. A higher CSAT score indicates greater customer satisfaction. This metric is commonly utilized across various sectors, including e-commerce, banking, and digital services, as an effective tool for assessing service quality and identifying areas for improvement.

III. RESULT AND DISCUSSION

The results demonstrate the performance of the BiLSTM model across three distinct feature extraction techniques, along with its integration with the Whisper ASR model for identifying harmful content in TikTok videos.

A. *Data Collection*

The data collection process resulted in a total of 15,941 labeled text samples used for training and evaluation. Of these, 13,057 instances were sourced from a public Kaggle dataset [16], and additionally, 2,884 tweets were collected from X or Twitter. These tweets were manually labeled to ensure accurate annotations.

B. *Data Labeling*

The manual labeling process resulted in a multi-label annotated dataset comprising six categories of harmful content: toxic, severe toxic, obscene, insult, threat, and identity hate. During the annotation process, several challenges were encountered, including ambiguous language, overlapping categories, and the subjectivity of certain labels such as "insult" and "toxic." Despite these challenges, consistent annotation guidelines helped maintain labeling quality.

A significant portion of the labeled data was found to fall under more than one category, affirming the multi-label nature of the task. For instance, it was common to observe data samples labeled both as toxic and insult, or obscene and severe toxic, highlighting the co-occurrence of harmful traits in online discourse. Table 4 displays a sample of the dataset utilized in this research, where the labels are represented as follows: a corresponds to toxic, b to severe toxic, c to obscene, d to insult, e to threat, and f to identity hate.

TABLE IV
DATASET SAMPLE

Text	Class Label					
	a	b	c	d	e	f
Penjajahan di atas bumi indonesia harus di hapus kan. Penjajahan ekonomi, budaya, hasil bumi oleh cina harus kita basmi. Demi NKRI. Usir cina	1	1	0	1	1	1
Kamu kok kaya jablay gitu maunya dijemput pake mobil',1,0,0,1,0,0	1	0	1	1	0	0
@ernestprakasa Memang bangsat yahudi ini	1	0	0	1	0	1
@neyaava @Ily_airaa Waspada hoaks pemecahan belah bangsa	0	0	0	0	0	0
@ns_IbnuMuhidin @msaid_didu Fufupapa pelacur bangsa	1	0	0	1	0	0

Table 4 presents five sample entries from the labeled dataset. Texts in which all label columns are assigned a value of 0 are considered non-harmful, whereas texts with at least one label column valued at 1 are classified as harmful content. The overall label distribution of the annotated dataset is shown in Table 5.

Table 5 illustrates the frequency distribution of each harmful content label in the dataset. The data exhibits a significant class imbalance, with the toxic class being the most prevalent (7,099 instances), whereas severe toxic and threat are notably underrepresented, with 600 and 392 instances, respectively. Furthermore, a total of 8,362 entries are not associated with any harmful content labels, indicating a substantial portion of neutral or non-harmful text within the dataset.

TABLE V
CLASS LABELS DISTRIBUTION

Class Label	Frequency
Toxic	7099
Severe Toxic	600
Obscene	887
Insult	2302
Threat	392
Identity Hate	2470

C. Data Preprocessing

The preprocessing stage effectively improved the quality and consistency of the textual data. Initial steps such as data cleaning, case folding, and noise removal standardized the input and eliminated irrelevant elements like URLs, emojis, and special characters. Text normalization and stemming reduced vocabulary complexity, while stopword removal filtered out non-informative words. The results of these preprocessing steps are presented in Table 6. All text was tokenized and padded to a fixed length of 60 tokens for uniform input to the model.

Three feature extraction methods learned embedding, Word2Vec, and FastText—were employed in this study. The learned embedding was trained jointly with the classification model using an embedding dimension of 32. In contrast, Word2Vec and FastText utilized pre-trained CBOW-based Indonesian embeddings with vector dimensions of 400 and 300, respectively. The resulting word embedding for each method is represented as an embedding matrix, where each row corresponds to the vector representation of a token in the vocabulary.

TABLE VI
CLASS LABELS DISTRIBUTION

Before Preprocessing	After Preprocessing
@txtdrimedia ANJ terus aja naik bangsat sat sat	anjing bangsat bangsat bangsat.
@heyooelii Lebih bangsat kau @heyooelii naikin kasus nyelipin iklan jualan. Musibah org lau manfaatin jadi ladang iklan	bangsat selip iklan jual musibah orang kamu manfaat ladang iklan
USER USER USER USER Bom yang real mudah terdeteksi bom yang terkubur suatu saat lebih dahsyat ledakannya itulah di sebut Revolusi Jiwa'	bom real mudah deteksi bom kubur dahsyat ledak revolusi jiwa
besok gue puasa jajan anjir awas aja ada yg ngajak gue jajan	besok puasa jajan anjir awas ajak jajan
@maxversai Iya betull awas nanti diajak tarohan sama yang kepalanya botak naek mustang item	iya awas ajak taruh kepala botak mustang hitam

D. Modeling

The hybrid deep learning model was developed by combining a BiLSTM-based text classification model with Whisper ASR. The BiLSTM model was trained using three different feature extraction techniques: learned embedding, Word2Vec, and FastText. Meanwhile, the Whisper ASR model was used without fine-tuning, utilizing the pre-trained “Small” variant.

1) *BiLSTM Modek*: The BiLSTM deep learning model was developed using three dataset split scenarios. The first scenario involved splitting the dataset into 60% for training, 20% for validation, and 20% for testing. The second scenario used and 70% training, 15% validation, and 15% testing. The third scenario used an 80% training, 10% validation, and 10% testing split. The details of both scenarios are presented in Table 7.

TABLE VII
SCENARIO SPLIT DATASET

Scenario	Split Dataset		
	Train	Validation	Test
I	60%	20%	20%
II	70%	15%	15%
III	80%	10%	10%

Based on the training experiments conducted under the three data split scenarios using three different feature extraction methods, Table 8 presents the types of models employed in this study.

TABLE VIII
MODELS’ PERFORMACES

Model	Feature Extraction
Model I	BiLSTM + Learned Embedding
Model II	BiLSTM + FastText
Model III	BiLSTM + Word2Vec

As shown in Table 8, this study utilizes three model types, each incorporating a different word embedding method. The performance results of each model are presented in Table 9.

TABLE IX
MODELS’ PERFORMACES

Model	Sc	Training		Validation		Epoch
		Loss	Rec	Loss	Rec	
Model I	**I**	0.0190	0.9752	0.0179	0.9743	75
	II	0.0210	0.9754	0.0189	0.9704	51
	III	0.0079	0.9797	0.0197	0.9632	69
Model II	I	0.0076	0.9907	0.0082	0.9921	46
	II	0.0070	0.9915	0.0059	0.9946	64
	III	0.0069	0.9915	0.0072	0.9876	45
Model III	I	**0.0057**	**0.9931**	0.0062	0.9894	53
	II	0.0085	0.9871	**0.0053**	**0.9962**	63
	III	0.0069	0.9926	0.0062	0.9884	35

Table 9 presents the training and validation performance of each model across the three data split scenarios outlined in

Table 8. The table reports three key aspects: loss, recall, and the number of training epochs at which early stopping occurred. Loss values provide an indication of the model's error during training and validation, while recall measures its ability to correctly identify relevant instances. The inclusion of early stopping epochs highlights the training efficiency and convergence behavior of each model, allowing for comparisons in both effectiveness and training stability. In general, all models achieved high recall scores across scenarios, indicating effective detection of harmful content.

Model I (using learned embedding) achieved its best validation recall of 0.9743 in Scenario I. Model II (FastText) showed consistently strong performance, reaching its highest validation recall of 0.9946 in Scenario II. Model III (Word2Vec) outperformed the others in Scenario II with the highest validation recall of 0.9962, along with relatively low validation loss, suggesting high accuracy and generalization capability.

Notably, Scenario II (70% train, 15% validation, 15% test) appears to offer a balanced split that supports optimal performance across models. These results suggest that both the choice of embedding method and data split ratio significantly influence model effectiveness.

TABLE X
MODEL TESTING

Model	Scenario	Recall	Hamming Loss	Total Parameters
Model I	**I**	**0.9786**	**0.0052**	**1,610,132**
	II	0.9702	0.0067	1,610,132
	III	0.9662	0.0074	1,610,132
Model II	I	0.9940	0.0023	12,997,988
	II	0.9917	0.0020	12,997,988
	III	0.9941	0.0015	12,997,988
Model III	I	0.9902	0.0017	17,247,188
	II	0.9918	0.0030	17,247,188
	III	0.9903	0.0021	17,247,188

Table 10 summarizes the testing performance of each model under the three different data split scenarios, using recall and Hamming Loss as evaluation metrics. The results demonstrate that all models perform well in detecting harmful content, with high recall values and low Hamming Loss across scenarios.

The BiLSTM model with FastText embeddings consistently outperformed the others, achieving the highest recall of 0.9941 and the lowest Hamming Loss of 0.0015 in Scenario III. This indicates superior ability to identify relevant instances while maintaining minimal label-wise misclassification. The Word2Vec-based model also performed competitively, particularly in Scenario II, where it achieved a recall of 0.9918. In contrast, the model using learned embeddings showed slightly lower performance, with its best recall of 0.9786 in Scenario I and gradually decreasing across other scenarios.

Although the BiLSTM model with FastText embeddings (Model II) in Scenario III achieved the highest testing performance, recording a recall of 0.9941 and a Hamming Loss of 0.0015, it comes with a significant trade-off in terms of computational complexity, with approximately 13 million parameters. In the context of a browser extension, where real-time and low-latency predictions are crucial for detecting harmful content, model efficiency becomes a critical consideration. High inference time due to large model size can hinder responsiveness and degrade user experience.

To address this, the model selection process considered not only predictive performance but also model size. The learned embedding-based model (Model I), despite slightly lower recall scores, maintains a relatively small footprint with only 1.6 million parameters, offering a better trade-off between accuracy and inference speed. Therefore, the final model was selected based on the best-performing configuration among models with lower parameter counts, ensuring both effective detection and real-time applicability in deployment environments such as browser extensions.

TABLE XI
CLASSIFICATION REPORT OF MODEL I SCENARIO I

Label	precision	recall	F1-score	Support
Toxic	**0.9921**	**0.9935**	**0.9928**	**1388**
Severe Toxic	0.9717	0.9537	0.9626	108
Obscene	0.9821	0.9880	0.9851	167
Insult	0.9612	0.9675	0.9643	461
Threat	0.9672	0.8429	0.9008	70
Identity Hate	0.9956	0.9681	0.9817	470
Micro avg	0.9853	0.9786	0.9819	2664
Macro avg	0.9783	0.9523	0.9645	2664
weighted avg	0.9853	0.9786	0.9818	2664

Table 11 presents the detailed classification report for Model I under Scenario I, showing precision, recall, and F1-score for each individual class. The results indicate that the model performs consistently well across most harmful content categories. High recall values are observed for labels such as Toxic (0.9935), Obscene (0.9880), and Insult (0.9675), suggesting the model is highly effective in correctly identifying these types of harmful content.

However, lower recall is noted for the Threat class (0.8429), indicating a relatively higher rate of false negatives in this category. This is a common challenge in multi-label classification when dealing with imbalanced datasets, particularly for minority classes such as Threat and Severe Toxic.

The micro-averaged recall and F1-score of 0.9786 and 0.9819, respectively, reflect the model's strong overall performance, while the macro average (recall: 0.9523, F1-score: 0.9645) highlights its ability to generalize across all classes regardless of support size. These results demonstrate that Model I under Scenario I offers a reliable balance between precision and recall, particularly for the most frequent and impactful harmful content categories. Figure 4 illustrates the confusion matrix of Model I under Scenario I, which was implemented in the browser extension.

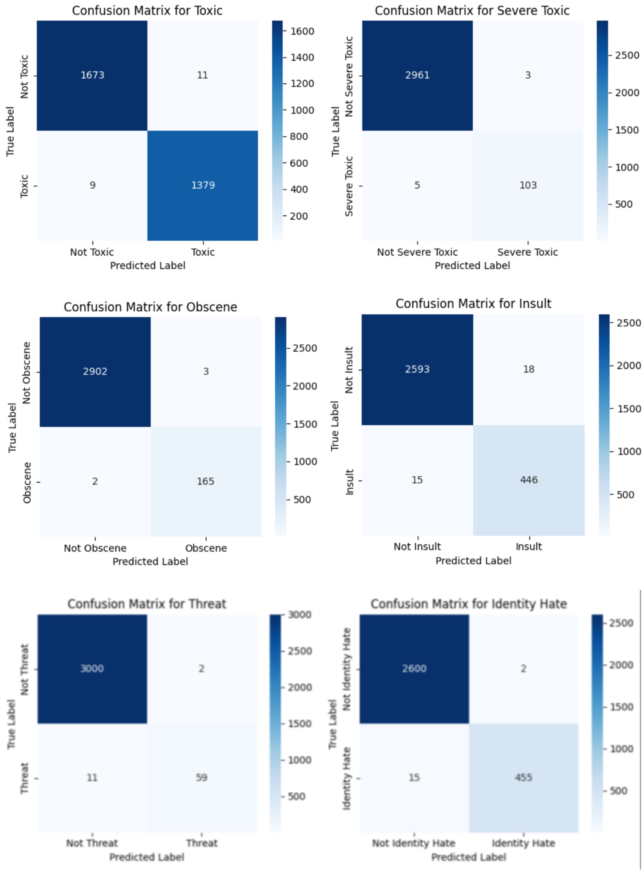


Figure 4. Confussion Matrix for Model 1 Scenario 1

Figure 4 displays the confusion matrices for each label in the multi-label classification task using Model I under Scenario I, as implemented in the browser extension. Each matrix provides a detailed view of true positives, true negatives, false positives, and false negatives for the six label categories: Toxic, Severe Toxic, Obscene, Insult, Threat, and Identity Hate.

The model shows strong performance across most categories, particularly for Toxic, Obscene, and Insult, where both true positives and true negatives dominate. For instance, in the Toxic category, the model correctly classified 1,379 toxic instances and 1,673 non-toxic instances, with only a small number of misclassifications.

In contrast, performance on Threat and Severe Toxic labels shows a relatively higher number of false negatives (e.g., 11 Threat instances misclassified as Not Threat, and 5 Severe Toxic instances as Not Severe Toxic), reflecting the challenge of identifying minority classes within imbalanced datasets.

Overall, the confusion matrices validate the model's effectiveness in distinguishing harmful content, while also highlighting areas particularly less frequent classes where further improvements may be necessary to enhance detection accuracy in real-world deployment.

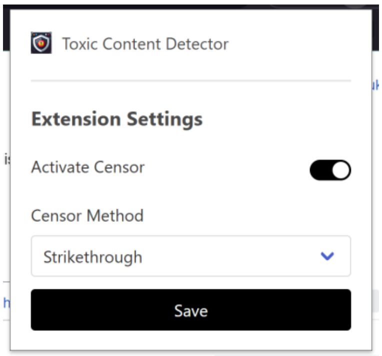
2) *Whisper ASR Model*: This study utilizes the Whisper ASR small model, which contains 244 million parameters. It offers relatively low memory usage, requiring up to only 2GB of VRAM and demonstrates significantly faster inference speed, being approximately four times faster than the large model variant. The selection of the Whisper small model was motivated by the need for efficient and rapid prediction of harmful content in TikTok videos, while maintaining a reasonably low word error rate (WER). To evaluate its performance, the Whisper small ASR model was tested on the The FLEURS dataset, short for Few-shot Learning Evaluation of Universal Representations of Speech for the Indonesian language, consisting of 687 audio samples. The model obtained a Word Error Rate (WER) of 41.64%, indicating a moderate level of transcription accuracy for Indonesian, which is acceptable for downstream tasks such as harmful content detection where general speech recognition suffices even in the presence of minor transcription errors.

E. Application Programming Interface

The application programming interface (API) developed using the FastAPI framework features two primary endpoints: /predict-webpage for detecting harmful content on general web pages, and /predict-tiktok-whisper for identifying such content in TikTok videos. Although both endpoints follow a similar detection workflow, they differ in input processing. For the /predict-webpage endpoint, Selenium is used to scrape text content from the target web page. The extracted text then undergoes preprocessing before being passed to the classification model, with the detection results returned via the API response. In contrast, the /predict-tiktok-whisper endpoint begins by downloading the audio from a TikTok video. The audio is then transcribed using the Whisper model, followed by text preprocessing and classification. The final prediction results are also delivered through the API response. The developed API then deployed using Google Cloud Platform's Cloud Run.

F. Browser Extension

The browser extension developed in this study incorporates two main features. The first feature is the ability to censor harmful textual content on web pages, with configurable settings as illustrated in Figure 5.


Toxic Content Detector

Extension Settings

Activate Censor ●

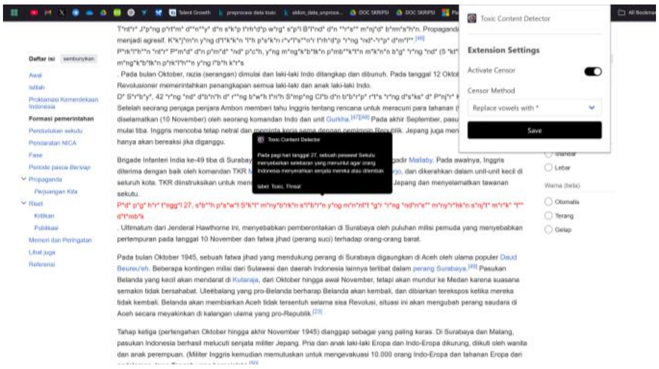
Censor Method

Strikethrough

Figure 5. Chrome Extension Settings

The extension offers five censorship methods: strikethrough, replace all characters with "X", replace vowels with "X", replace all characters with asterisks (), and replace vowels with asterisks (), with the default method set to "replace vowels with *". The second feature provides safety notifications for TikTok videos currently being accessed by the user, alerting them when potentially harmful content is detected.

The extension censors potentially harmful words based on the user's selected settings. The censorship results produced by the Chrome extension are illustrated in Figure 6. In addition to censoring harmful sentences, the extension also displays an informative tooltip containing the prediction label when the censored sentence is hovered over, providing contextual information about the detected content.



Toxic Content Detector

Extension Settings

Activate Censor: on

Censor Method: Replace vowels with *

Save

T*nt*r* J*p*ng p*rt*m* d**n**y* d*n s*k*p t*rh*d*p w*rg* s*p*l B*l*nd* d*n **r*s** m*nj*d* b*m*s*h*n. Propaganda [illegible] menjadi agresif. K*k*j*m*n y*ng d*l*k*k*n *l*h p*s*k*n r*v*l*s**n*r t*rh*d*p *r*ng *nd*-*r*p* d*m*l**.[46]

P*rk*l*h**n *nt*r* P*m*d* d*n p*m*d* *nd* p*c*h, y*ng m*ng*k*b*tk*n p*mb**k*l*n m*k*n*n b*g* *r*ng *nd* (5 *k* [illegible] m*ng*k*b*tk*n p*rk*l*h**n y*ng l*b*h k*r*s

. Pada bulan Oktober, razia (serangan) dimulai dan laki-laki Indo ditangkap dan dibunuh. Pada tanggal 12 Okto[illegible] Revolusioner memerintahkan penangkapan semua laki-laki dan anak laki-laki Indo.

D* S*r*b*y*, 42 *r*ng *nd* d*b*n*h d* r**ng b*w*h t*n*h S*mp*ng Cl*b d*n b*b*r*p* r*t*s *r*ng d*s*ks* d* P*nj*r* [illegible] Setelah seorang penjaga penjara Ambon memberi tahu Inggris tentang rencana untuk meracuni para tahanan [illegible] diselamatkan (10 November) oleh seorang komandan Indo dan unit Gurkha.[47][48] Pada akhir September, pasu[illegible] mulai tiba. Inggris mencoba tetap netral dan meminta kerja sama dengan pemimpin Republik. Jepang juga men[illegible] hanya akan bereaksi jika diganggu.

Brigade Infanteri India ke-49 tiba di Surabay[illegible] gadir Mallaby. Pada awalnya, Inggris diterima dengan baik oleh komandan TKR [illegible] rjo, dan dikerahkan dalam unit-unit kecil di seluruh kota. TKR diinstruksikan untuk men[illegible] Jepang dan menyelamatkan tawanan sekutu.

Toxic Content Detector

Pada pagi hari tanggal 27, sebuah pesawat Sekutu menyebarkan selebaran yang menuntut agar orang Indonesia menyerahkan senjata mereka atau ditembak

label: Toxic, Threat

P*d* p*g* h*r* t*ngg*l 27, s*b**h p*s*w*t S*k*t* m*ny*b*rk*n s*l*b*r*n y*ng m*n*nt*t *g*r *r*ng *nd*n*s** m*ny*r*hk*n s*nj*t* m*r*k* *t** d*t*mb*k

. Ultimatum dari Jenderal Hawthorne ini, menyebabkan pemberontakan di Surabaya oleh puluhan milisi pemuda yang menyebabkan pertempuran pada tanggal 10 November dan fatwa jihad (perang suci) terhadap orang-orang barat.

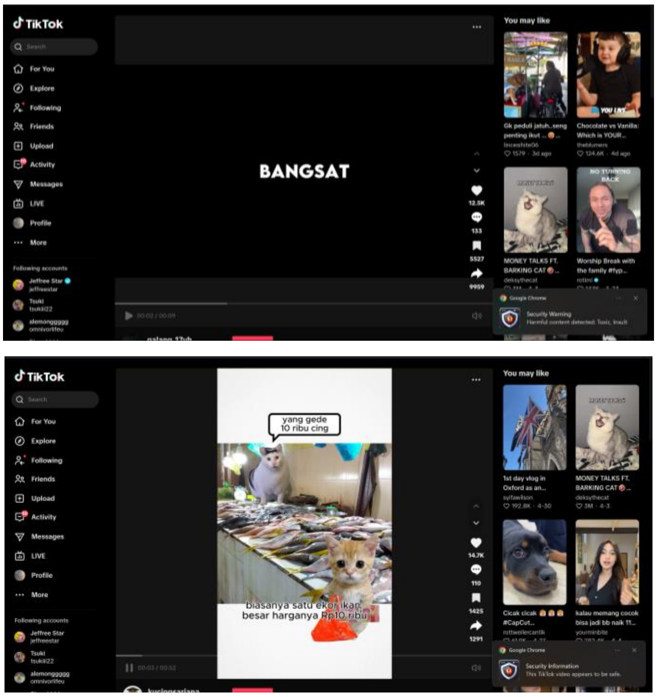
Pada bulan Oktober 1945, sebuah fatwa jihad yang mendukung perang di Surabaya digaungkan di Aceh oleh ulama populer Daud Beureu'eh. Beberapa kontingen milisi dari Sulawesi dan daerah Indonesia lainnya terlibat dalam perang Surabaya.[49] Pasukan Belanda yang kecil akan mendarat di Kutaraja, dari Oktober hingga awal November, tetapi akan mundur ke Medan karena suasana semakin tidak bersahabat. Uleëbalang yang pro-Belanda berharap Belanda akan kembali, dan dibiarkan terekspos ketika mereka tidak kembali. Belanda akan membiarkan Aceh tidak tersentuh selama sisa Revolusi, situasi ini akan mengubah perang saudara di Aceh secara meyakinkan di kalangan ulama yang pro-Republik.[23]

Tahap ketiga (pertengahan Oktober hingga akhir November 1945) dianggap sebagai yang paling keras. Di Surabaya dan Malang, pasukan Indonesia berhasil melucuti senjata militer Jepang. Pria dan anak laki-laki Eropa dan Indo-Eropa dikurung, diikuti oleh wanita dan anak perempuan. (Militer Inggris kemudian memutuskan untuk mengevakuasi 10.000 orang Indo-Eropa dan tahanan Eropa dari [illegible]

Figure 6. Result of Harmful Sentence Censorshi

The harmful content detection feature for the TikTok platform functions by sending security notifications based on the content of the video. There are three types of notifications, as illustrated in Figure 7. The extension displays the message “This TikTok video appears to be safe” when no harmful sentences are detected. If harmful content is identified, the notification “Harmful content detected” is shown, along with the classification label of the video. Additionally, the message “Can’t detect this TikTok video at the moment” is displayed when the detected video is not in the Indonesian language.

When the Chrome extension is activated while a TikTok video is being accessed, the video will automatically pause. If the security notification indicates that the video contains harmful content, the video will remain paused. However, if the video is deemed safe, it will resume playing automatically.



Security Warning
Harmful content detected: Toxic, Insult

Security Information
This TikTok video appears to be safe.

Figure 7. Security Notification Results for Harmful Content on TikTok

G. Browser Extension CSAT Score

This study conducted three types of evaluations on the developed browser extension: user experience testing, performance evaluation, and detection relevance assessment. The evaluations involved 22 participants. The user experience test yielded a Customer Satisfaction Score (CSAT) of 95.45%, indicating a high level of satisfaction with the extension's usability. The performance evaluation of the extension resulted in a CSAT score of 79.55%, suggesting that the harmful content detection functionality for both webpages and TikTok videos performs well. The detection relevance test on webpages achieved a high satisfaction score of 90.91%, showing that most users agreed with the harmful sentences flagged and censored. However, the CSAT score for the relevance of TikTok video safety notifications was lower at 77.27%. This discrepancy may be attributed to the Whisper model's relatively high word error rate (WER) of 41.64% when evaluated on the FLEURS dataset, despite the strong performance of the BiLSTM classification model. Overall, the average CSAT score for detection relevance across both features was 84.09%, indicating that the extension’s detection accuracy is considered very good by users.

In contrast to prior studies that focus exclusively on text-based detection [11], [12], [13] or are limited to offline model evaluation [15], this research presents a novel contribution through the real-time deployment of a hybrid deep learning system consisting of the Whisper ASR model and a BiLSTM classifier within a browser extension. Unlike multilingual or general-purpose detection systems, this study specifically targets harmful content in the Indonesian language, addressing a relatively underexplored area in harmful content detection research. The originality of this work lies in its integration of audio transcription and multi-label classification into a lightweight, real-time detection system capable of handling both textual webpages and video-based content from platforms such as TikTok. While previous research has demonstrated the effectiveness of BiLSTM in toxic content classification, its real-world implementation particularly in the form of a browser extension remains limited. By embedding the detection system into a Chrome extension, this study advances the field by bridging the gap between academic model development and practical end-user applications. Furthermore, the model is trained to recognize six distinct labels, enabling more granular classification than binary or ternary schemes used in earlier studies. The evaluation includes not only model performance metrics but also user-centered assessments such as CSAT scores, providing a holistic understanding of the system's practical utility and acceptance. This combination of a hybrid architecture, real-time functionality, and direct user feedback distinguishes this research from prior work and demonstrates its potential for scalable deployment in digital safety initiatives.

IV. Conclusion

This study aims to detect harmful content on web pages through the implementation of a browser extension that integrates a hybrid deep learning approach, demonstrating the effectiveness of a BiLSTM deep learning model integrated with various feature extraction techniques and browser-based implementation for detecting harmful content. Among the three tested feature extraction methods: learned embeddings, FastText, and Word2Vec. The FastText-based BiLSTM model consistently outperformed others in terms of recall and Hamming Loss, particularly under Scenario III. However, despite its superior predictive performance, the FastText model's high computational cost (with approximately 13 million parameters) presents challenges for real-time deployment in resource-constrained environments such as browser extensions. As a trade-off between performance and efficiency, the final deployed model was Model I under Scenario I, which utilized learned embeddings. This configuration maintained competitive recall (0.9786) and hamming loss (0.0052), with a significantly smaller parameter size (1.6 million), making it more suitable for real-time use in browser-based applications.

Further evaluation of Model I under Scenario I revealed high performance in detecting common harmful categories such as Toxic, Obscene, and Insult, while detection of minority classes like Threat and Severe Toxic remains challenging due to class imbalance. The confusion matrix analysis confirmed the model's strength in minimizing false positives and negatives for the majority classes, although some misclassifications still occurred for the less frequent ones.

The integration of the BiLSTM model into a browser extension enabled real-time harmful content filtering on web pages using five customizable censorship methods: strikethrough, replacing all characters with "X," replacing vowels with "X," replacing all characters with asterisks (*), and replacing vowels with asterisks (*). Additionally, the combination of the BiLSTM model with Whisper ASR facilitated audio-based content analysis on TikTok, allowing the extension to issue contextual safety notifications based on video transcriptions

User satisfaction with the Chrome extension was evaluated through Customer Satisfaction Score (CSAT) surveys across three dimensions: user experience, performance, and detection relevance. The extension achieved an Excellent CSAT rating of 95.45% in user experience and functionality. Although the performance evaluation received a slightly lower CSAT score of 79.55%, it still fell within the Good CSAT category. The detection relevance component earned an overall CSAT score of 84.09%, indicating a high level of user approval regarding the accuracy and reliability of harmful content identification.

For future research, alternative deep learning architectures such as transformer-based models could be explored for harmful content detection, as they may offer improved contextual understanding and better generalization. Additionally, fine-tuning the Whisper ASR model on domain-specific or language-specific datasets, particularly in Indonesian, has the potential to reduce the current word error rate (WER) and enhance the accuracy of transcriptions used for downstream classification tasks. To further improve performance, especially in handling imbalanced datasets, future work may consider implementing data augmentation strategies such as oversampling minority classes or under sampling majority classes. These approaches could help mitigate performance degradation due to class imbalance and lead to more robust and equitable harmful content detection across diverse input data.

References

[1] S. Kemp, "Digital 2024: Indonesia." Accessed: Jun. 12, 2025. [Online]. Available: https://datareportal.com/reports/digital-2024-indonesia

[2] N. Rahman, "Social Media, Freedom of Expression and Right to Privacy: An Analysis," *SSRN Electron. J.*, 2023, doi: 10.2139/ssrn.4637111.

[3] UNICEF, "What is harmful content?," UNICEF Australia. Accessed: Jun. 12, 2025. [Online]. Available: https://www.unicef.org.au/parent-teacher-resources/online-safety/harmful-content?srsltid=AfmBOoqNoyHkIR5V8Xwm5EiFpdX3Hxw0J3O39c-agBUN2g5Oe3LDwoLf

[4] C. Febriyani, "Bahaya Ujaran Kebencian di Dunia Maya Diatur Sebagai Tindak Pidana di UU ITE," Industry.co.id. Accessed: Jun. 12, 2025. [Online]. Available: https://www.industry.co.id/read/93219/bahaya-ujarankebencian-di-dunia-maya-diatur-sebagai-tindak-pidana-di-uuite

[5] Pusiknas Bareskrim Polri, "Lima Hari, Belasan Polda Tangani Kasus Pencemaran Nama Baik." Accessed: Jun. 12, 2025. [Online]. Available:

https://pusiknas.polri.go.id/detail_artikel/lima_hari,_belasan_polda_tangani_kasus_pencemaran_nama_baik

[6] R. P. Utami, “Web Page adalah Bagian Terpenting Website, Benarkah? | Bamaha Digital,” https://bamahadigital.com/. Accessed: Jun. 12, 2025. [Online]. Available: https://bamahadigital.com/web-page-adalah/

[7] M. Rosyida, “5++ Perbedaan Web Page dan Web Site yang Perlu Kamu Tahu.” Accessed: Jun. 12, 2025. [Online]. Available: https://www.domainesia.com/berita/perbedaan-web-page-dan-web-site/

[8] S. Kemp, “TikTok Users, Stats, Data, Trends, For 2025,” DataReportal – Global Digital Insights. Accessed: Jun. 12, 2025. [Online]. Available: https://datareportal.com/essential-tiktok-stats

[9] C. Janiesch, P. Zschech, and K. Heinrich, “Machine learning and deep learning,” *Electron. Mark.*, vol. 31, no. 3, pp. 685–695, Apr. 2021, doi: 10.1007/s12525-021-00475-2.

[10] A. Raup, W. Ridwan, Y. Khoeriyah, S. Supiana, and Q. Zaqiah, “Deep Learning dan Penerapannya dalam Pembelajaran,” *JIIP - J. Ilm. Ilmu Pendidik.*, vol. 5, pp. 3258–3267, Sep. 2022, doi: 10.54371/jiip.v5i9.805.

[11] A. Perwira Joan Dwitama, D. Fudholi, and S. Hidayat, “Indonesian Hate Speech Detection Using Bidirectional Long Short-Term Memory (Bi-LSTM),” *J. RESTI Rekayasa Sist. Dan Teknol. Inf.*, vol. 7, pp. 302–309, Mar. 2023, doi: 10.29207/resti.v7i2.4642.

[12] M. Neog and N. Baruah, “A hybrid deep learning approach for Assamese toxic comment detection in social media,” *Procedia Comput. Sci.*, vol. 235, pp. 2297–2306, 2024, doi: https://doi.org/10.1016/j.procs.2024.04.218.

[13] S. Kaur, S. Singh, and S. Kaushal, “Deep learning-based approaches for abusive content detection and classification for multi-class online user-generated data,” *Int. J. Cogn. Comput. Eng.*, vol. 5, pp. 104–122, 2024, doi: https://doi.org/10.1016/j.ijcce.2024.02.002.

[14] E. Zahra, Y. Sibaroni, and S. Prasetyowati, “Classification of Multi-Label of Hate Speech on Twitter Indonesia using LSTM and BiLSTM Method,” *JINAV J. Inf. Vis.*, vol. 4, pp. 170–178, Jul. 2023, doi: 10.35877/454RI.jinav1864.

[15] J. An, W. Lee, Y. Jeon, J. Ok, Y. Kim, and G. Lee, “An Investigation into Explainable Audio Hate Speech Detection,” Jan. 2024, pp. 533–543. doi: 10.18653/v1/2024.sigdial-1.45.

[16] Rivaldo, “aldon_data_unprocessed.” Accessed: Jun. 13, 2025. [Online]. Available: https://www.kaggle.com/datasets/aldonistan/aldon-data-unprocessed

[17] I. Kurniawan and A. Susanto, “Implementasi Metode K-Means dan Naïve Bayes Classifier untuk Analisis Sentimen Pemilihan Presiden (Pilpres) 2019,” *Eksplora Inform.*, vol. 9, pp. 1–10, Sep. 2019, doi: 10.30864/eksplora.v9i1.237.

[18] I. N. Bayu, I. M. A. D. Suarjaya, and P. Buana, “Classification of Indonesian Population’s Level Happiness on Twitter Data Using N-Gram, NaÃ¯ve Bayes, and Big Data Technology,” *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 12, p. 1944, Oct. 2022, doi: 10.18517/ijaseit.12.5.14387.

[19] R. I. Pristiyanti, M. A. Fauzi, and L. Muflikhah, “Sentiment Analysis Peringkasan Review Film Menggunakan Metode Information Gain dan K-Nearest Neighbor,” *J. Pengemb. Teknol. Inf. Dan Ilmu Komput.*, vol. 2, no. 3, pp. 1179–1186, 2017.

[20] M. Susanty and S. Sukardi, “Perbandingan Pre-trained Word Embedding dan Embedding Layer untuk Named-Entity Recognition Bahasa Indonesia,” *Petir*, vol. 14, pp. 247–257, Sep. 2021, doi: 10.33322/petir.v14i2.1164.

[21] M. Suri, “A Dummy’s Guide to Word2Vec. Essentials of Word2Vec.” Accessed: Jun. 12, 2025. [Online]. Available: https://medium.com/@manansuri/a-dummys-guide-to-word2vec-456444f3c673

[22] T. Mikolov, K. Chen, G. s Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” *Proc. Workshop ICLR*, vol. 2013, Jan. 2013.

[23] M. S. Jahan and M. Oussalah, “A systematic review of Hate Speech automatic detection using Natural Language Processing,” May 2021, doi: 10.48550/arXiv.2106.00742.

[24] A. Girsang, “Word Embedding dengan FastText.” Accessed: Jun. 12, 2025. [Online]. Available: https://mti.binus.ac.id/2021/12/31/word-embedding-dengan-fasttext/

[25] S. Cornegruta, R. Bakewell, S. Withey, and G. Montana, “Modelling Radiological Language with Bidirectional Long Short-Term Memory Networks,” Sep. 2016, doi: 10.48550/arXiv.1609.08409.

[26] A. Graves, S. Fernández, and J. Schmidhuber, “Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition.,” Jan. 2005, pp. 799–804.

[27] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, “Convolutional neural networks for time series classification,” *J. Syst. Eng. Electron.*, vol. 28, pp. 162–169, Feb. 2017, doi: 10.21629/JSEE.2017.01.18.

[28] A. Vaswani *et al.*, “Attention Is All You Need,” Jun. 2017, doi: 10.48550/arXiv.1706.03762.

[29] A. Radford, J. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, “Robust Speech Recognition via Large-Scale Weak Supervision,” Dec. 2022, doi: 10.48550/arXiv.2212.04356.

[30] S. Lee, “Hamming Loss Explained: Key Insights for Multi-label Learning.” Accessed: Jul. 03, 2025. [Online]. Available: https://www.numberanalytics.com/blog/hamming-loss-explained-key-insights