# UAEMemex Participation at Dimemex 2025: Exploring Lexical and Semantic Information to Detect Hate, Inappropriate, and Harmless Memes

Verónica Neri-Mendoza[1,2*,†], Jonathan Rojas-Simon[2,*,†], Yulia Ledeneva[2,*,†], Yorne Alejandrina Santos-Bobadilla[2], Abner Alain Gil-García[2], Ángel Baron-García[2], René Arnulfo Garcia-Hernández[2]

[1] *Secretariat of Science, Humanities, Technology and Innovation, 1582 Insurgentes Sur Avenue, Crédito Constructor Benito Juárez, Mexico City, Mexico.*

[2] *Autonomous University of the State of Mexico, Instituto Literario 100, Toluca 50000, México*

**Abstract**

As part of the IberLef 2025 workshop, this paper presents the framework developed by the UAEMemex team for the subtask of classifying hate speech, inappropriate content, and harmless content in Spanish-language memes from Mexico, as part of the DIMEMEX-2025 competition. Given the growing dissemination and social impact of hate speech and inappropriate content on social media, research in Natural Language Processing (NLP) for their automatic detection has gained relevance. The complexity of identifying these categories, particularly in the meme format that fuses lexical and visual information, requires sophisticated approaches. For this reason, we explored lexical information through ASCII vectorization and semantics through vector representations obtained with BERT and Doc2Vec to discern distinctive patterns among the three categories. Finally, these representations were used as input for the Logistic Regression, Multilayer Perceptron (MLP) and KNN algorithms for classification.

**Keywords**

Memes, Hate Speech, Inappropriate Content, ASCII, BERT, Doc2Vec, MLP, KNN, and LR

## 1. Introduction

This paper presents a framework of classification used by the UAEMemex team in Subtask 1: Detection of hate speech, inappropriate content, and harmless memes. Participants were free to use any approach of their choice, for the detection of inappropriate memes from Mexico. (DIMEMEX-2025) [1], at the seventh workshop of the Iberian Language Evaluation Forum (IberLef 2025)[2].

The scientific study of hate speech and inappropriate content on multimodal social media, from a Natural Language Processing (NLP) perspective, has an unquestionable potential for social impact. With the rapid development of mobile and web technologies, such content has become increasingly widespread on social media platforms, as it is easy to publish any opinion. It has been frequently observed that user conversations often veer into inappropriate areas, such as insults and rude and impolite comments about individuals or certain groups or communities [3].

Recent studies confirm that exposure to these contents has serious offline consequences for historically disadvantaged communities. Thus, research on the automatic detection of hate speech and inappropriate content has attracted significant attention. Determining the presence of hate

*IberLEF 2025, September 2025, Zaragoza, España*
1 Corresponding author.
† These authors contributed equally.

veronica.nerimendoz@gmail.com (V. Neri-Mendoza); jrojass@uaemex.mx (J. Rojas-Simón); ynledeneva @uaemex.mx (Y. Ledeneva); ysantosb001@alumno.uaemex.mx (Y. A. Santos-Bobadilla); agilg003@alumno.uaemex.mx (A. A. Gil-García); abarong001@alumno.uaemex.mx (A. Barón-García); reagarciah @uaemex.mx (R.A. García-Hernández)
0000-0002-9582-9502 (V. Neri-Mendoza); 0000-0002-0389-4201 (J. Rojas-Simon); 0000-0003-0766-542X (Y. Ledeneva); 0000-0001-7941-377X (R.A. García-Hernández)

speech and inappropriate content in a multimodal format is not straightforward, even for human interpretation.

In the meme format it poses significant challenges for automated moderation and semantic understanding. While memes often link complex meanings through the interaction of textual and visual elements, their potential to spread hate speech and inappropriate material demands the development of methodologies for their detection.

Our proposal focused on exploring lexical information through ASCII vectorization and semantic information in the textual components of memes through BERT (Bidirectional Encoder Representations from Transformers) vectorization. We proposed an in-depth analysis of lexical items and their contextual meaning through semantic models that can provide patterns to differentiate hate speech, inappropriate content, and memes without any such content. In addition, we implemented Logistic Regression, MLP, and KNN algorithms with the information expressed in text format.

This paper is organized as follows: Section 2 presents the relevant state-of-the-art works that address hate speech and inappropriate content detection; On the other hand, Section 3 describes the proposed approach; While Section 4 presents the empirical results derived from the application of the approach; and, finally, Section 5 presents the conclusions.

## 2. State-of-the-art works

Regarding BERT vectorization, this vectorization has been used in some works since BERT analyzes words considering the complete context surrounding them (both the words before and after them). This vectorization allows to understand the meaning of the words. Because of this, this type of representation has been used in several research, such as in [4] that address the automatic identification of hate speech in social media was investigated. To address the challenge of a lack of labeled data and classification biases, a transfer learning strategy based on BERT, a pre-trained language model, was proposed. It is ability to capture the context of offensive language was evaluated using fine-tuning techniques, and its performance was compared with previous approaches using public datasets tackling topics such as racism, sexism, and offensive content on Twitter (currently known as X). The results indicated that the model achieved significant improvements in precision and recall. In addition to revealing biases in annotation and data collection, which it is advisable to use to realize the potential of a more accurate model.

Moreover, hate speech detection on social media was investigated in [5], focusing on content related to religion, race, gender, and sexual orientation. To capture the semantics of the language used in these speeches, the BERT language model was used as a vectorization method. The results showed that BERT achieved an F1-score of up to 96% on a balanced dataset, outperforming other methods such as LSTM neural networks with domain-specific embeddings, which achieved an F1-score of 93%.

On the other hand, lexical features have proven to be fundamental in various NLP tasks, as they allow the representation of crucial information from documents for applications such as clustering and classification. In the specific field of text classification, character, or word-level, n-grams constitute one of the most widely used representations. However, their inherent high dimensionality entails considerable computational costs. Given this situation in mind, our objective focuses on identifying a lower-dimensional representation capable of capturing the stylistic and lexical particularities of a document. Consequently, the probability of occurrence of each character in its ASCII encoding within the text was calculated [6], [7].

In [8], the authors developed a hate speech content detection system, based on Twitter posts, using K-Nearest Neighbor (KNN) method, in conjunction with the TF-IDF feature extraction technique. The main objective was to identify potential violations of the Indonesian Electronic Information and Transaction Act (UU ITE). Standard data preprocessing techniques were applied, and TF-IDF was used to convert texts into numerical vectors. Classification was performed using KNN (K=10), achieving an accuracy of 67.86% using the cosine distance metric. In an evaluation with 100 new tweets, the accuracy reached 77%, validated by UU ITE law experts. The study concludes that KNN, combined with TF-IDF, is an effective tool for hate speech detection, although its accuracy could be improved depending on more balanced data and advanced natural language processing techniques.

In the work done in [9], it focused on the classification of Indonesian-language hate speech text, using an improved version of the KNN algorithm. To improve the representation of text features, a term weighting technique called TF-IDF-ICSρF was applied. The results indicated that the combination of TF-IDF-ICSρF with the improved KNN algorithm achieved an average accuracy of 88.11%, significantly outperforming the traditional KNN approach with TF-IDF, which achieved an accuracy of 70.30%.

In [10] the detection of hate speech on Twitter was explored using NLP and machine learning techniques. A logistic regression model was proposed to classify tweets into three categories: hate speech, offensive language, and none of the above. Standard preprocessing techniques were applied, and TF-IDF was used for text vectorization. The results demonstrated that the model achieved an accuracy of 93%, highlighting its effectiveness in automatically identifying harmful content on social media. The study highlights the usefulness of logistic regression combined with NLP representations as an effective solution for content moderation on digital platforms.

## 3. Proposed Approach

This section digests the proposed approach's overview. First, we explain the data stratification on which our proposed approach was tested. Second, we describe how the data was preprocessed. Third, we describe the text representation models employed for each meme. Finally, we describe the classifiers we used and their hyperparameters.

### 3.1. Data Stratification

Regarding the subtasks of DIMEMEX-2025, the proposed approach was tested on Subtask 1 (Detection of Hate Speech, Inappropriate, and Harmless Memes). In the first phase, the task organizers released training and development sets, providing only the labels in the training set to create classification models. The labels in the development set were not available, but proposed approaches could be evaluated with these data through the CodaLab platform [11].

On the other hand, the data stratification of the training set is shown in . As observed, it consists of 2263 Mexican Spanish memes from social media, of which 62.08% belong to the class “Harmless”, 20.86% belong to the class "Inappropriate content", and 17.06% belong to the "Hate speech" class (see table 1). With this information in mind, we notice that the distribution of memes is disbalanced, providing most of the memes in the class "Harmless". This situation represents a challenge for participants to distinguish any aggression.

**Table 1**
Distribution of samples in the training set.

| Class | # Samples | Percentage |
|---|---|---|
| Hate speech | 386 | 17.06% |
| Inappropriate content | 472 | 20.86% |
| Harmless | 1405 | 62.08% |
| Total | 2263 | 100.00% |

In addition, it is worth mentioning that the information for each meme is organized according to the examples shown in Figure 1. Each meme has its image, OCR text, image caption or description, and class. For the proposed approach, we focused on analyzing the OCR texts because we assumed that they contain more specific information that may be related to hate speech and inoffensive content.

| | | | |
|---|---|---|---|
| Image | | | espues de los memes racistas y xenofobos seguimos siendo hermanos<br>Porque somo seres civilizados |
| Text | NO ESTÁ MAL | -Pero esa mujer te hizo brujería!<br>-la brujería... | Después de los memes racistas y xenófobos, seguimos siendo hermanos. Porque somos seres civilizados. |
| Description | La imagen es un meme que presenta un primer plano de un hombre con piel oscura y cabello corto. Tiene una expres ... | La imagen es un meme que presenta un formato de dos paneles extraídos de una escena de una película o serie. En el primer panel ... | La imagen es un meme que presenta una serie de personajes con cabezas de diferentes criaturas o personas super-puestas sobre figuras de palitos, cada una vestida con camisetas que representan banderas de varios países latinoamerica |
| *Class* | *Harmless* | *Inappropriate content* | *Hate speech* |

**Figure 1:** Examples of memes per class.

### 3.2. Pre-processing

After extracting OCR text memes, we filtered them to eliminate unnecessary information. This process involves the following steps:

Tokenization: All words were tokenized and separated from non-alphanumeric characters and punctuation marks to improve the association or "understanding" of words for each meme.

Normalization: We removed non-alphanumeric characters and punctuation marks from tokenized texts, obtaining words that were then converted to lowercase.

Stopwords removal: Finally, normalized texts were introduced to stopword removal, eliminating common low-meaning words (*e.g.*, *de*, *la*, *que*, *el*). In particular, we removed all stopwords from each text through the Spanish stopwords list provided by the NLTK toolkit (https://bit.ly/2DqKPvW).

In previous studies on hate speech detection, the steps mentioned above have been considered standard procedures to create classification models. However, the proposed approach does not necessarily rely on these methods. Below, we provide a detailed explanation of its structure.

### 3.3. Model Architecture

The model architecture is based on the framework of components we call NLPClassKit. NLPClassKit is an assembly of software components that allows classification models to be developed for NLP projects. It is currently available in a GitLab repository (https://gitlab.com/JohnRojas/NLPClassKit), and it considers an architecture of processes according to Figure 2.

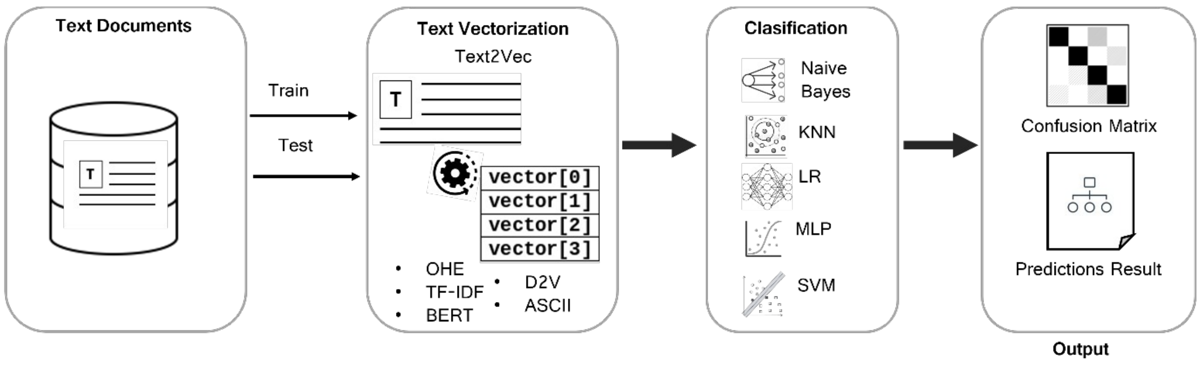

**Figure 2:** Architecture of NLP ClassKit.

### 3.3.1. Text vectorization

Once text documents were extracted or preprocessed, they were used as input for text vectorization methods. Such methods are essential for extracting relevant text information at different levels (*e.g.*, lexical or semantic). Some of these methods are incorporated into a software component called Text2Vec. Text2Vec is a component developed in Python and available in a GitLab repository (https://gitlab.com/MLComponents1/Text2Vec) that encodes texts according to the following techniques:

*One-hot-encoding (OHE):* OHE is a straightforward method for representing words as binary vectors. It generally creates a vector vocabulary from input text, where each word is assigned a unique index and a binary vector is created for each word (*i.e.*, it assigns 1 to the index of the word appearing in the document and zero elsewhere) [12]. Concerning other methods, OHE does not require high computational resources but generates high-dimensional vectors.

*TF-IDF (Term Frequency-Inverse Document Frequency):* It is a frequency-based method for evaluating the importance of a word to a document in a collection (corpus). It combines Term-Frequency (how often a word appears in a document) and Inverse Document Frequency (How unique or rare the word is across all documents).

*Doc2Vec (D2V):* It is a Word2Vec-based method that generates vector representations for entire documents, paragraphs, or sentences instead of individual words [13]. In general, it captures the context and semantics of a document in a fixed-length numerical vector, allowing for comparison, classification, or clustering of documents.

*BERT (Bidirectional Encoder Representations from Transformers):* It is a pre-trained method based on attention mechanisms that understands the context of words in a sentence by looking at both the left and right sides (bidirectional) [14]. Unlike the methods mentioned above, BERT uses a transformer architecture to deeply understand meaning and relationships in language, creating contextual vectors. Moreover, it is fine-tuned with specific-domain text data, and it does not require preprocessing steps mentioned in the Section 3.2.

*ASCII:* This method proposed in [6], [7] that considers the frequency of characters to determine the probability that each character may appear from a given OCR text. Formally, it is shown in Eq. (1):

$$ASCII(d)=\left[p(c_1),p(c_2),\ldots,p(c_{255})\right],\ p(c_i)=\frac{f(c_i)}{len(d)}, \tag{1}$$

where the function $ASCII(d)$ receives the input OCR text $d$. Next, it generates a vector representation of 255 values; each $p(c_i)$ represents the probability that the character $c_i$ appears in $d$. Such probabilities are calculated by dividing the frequency of character $c_i$ ($f(c_i)$) and $len(d)$ (a function that counts the number of characters). Furthermore, the value $p(c_{256})$ is used for emojis or unknown characters that may appear in the same OCR text.

### 3.3.2. Classification

The vectors obtained from the previous step are used as input to a set of supervised machine learning algorithms, which are briefly described as follows:

*NaiveBayes (NB):* The NB algorithm relies on the Bayes theorem, which deals with probability calculus. Considering its input features, the NB computes the probability that a text may belong to a specific class.

*K-Nearest Neighbors (KNN):* KNN is a well-known machine learning algorithm that operates on the majority rule principle. It predicts the label of a test data point by assigning it to the class most common among its $K$ nearest training data points in the feature space.

*Logistic Regression (LR):* The LR is a supervised machine learning algorithm that measures the relationship between a set of features $(X)$ and a binary output $(y \in \{0,1\})$. Mathematically, LR is expressed as $1/(1+e^{-z})$, where $z$ is the linear combination between the input features $(x_i)$ and model parameters $(w_i)$. The output values range from $0$ to $1$, indicating the likelihood that the input belongs to the output $1$.

*Multilayer Perceptron (MLP):* The MLP is a deep learning model composed of three types of fully connected layers: (i) the input layer, (ii) one or more hidden layers, and (iii) the output layer. Furthermore, this model can capture complex data relations and solve text classification tasks.

*Support Vector Machines (SVM):* SVM is a supervised machine learning algorithm for classification tasks. It works by finding the optimal hyperplane that best separates data points of different classes in a high-dimensional space. The key idea is to maximize the margin between the nearest points (support vectors) of different classes.

### 3.3.3. Generation and interpretation of results

Finally, the result of the classification process is given in the last stage, where each algorithm described in the previous section generates the following information:

- Confusion matrix: Shows the number of texts classified correctly/incorrectly.
- Predictions: Shows the prediction labels generated in the training stage of each algorithm.
- Performance of generated models: Shows a detailed description of the performance of each classification model in terms of Recall, Precision, F1-score, macro F1-score, and Accuracy.

## 4. Results

Table 2 shows the best four results obtained by our team. The values obtained for the F1 measure, Precision, Recall, as well as the method used to obtain these results are shown: LR, MLP and KNN. Finally, the vectorization column shows whether the obtained result was based on BERT, D2V or ASCII representations.

**Table 2.**
Best results obtained by the team. The boldface number indicates the best result.

| Team | F1 | Precision | Recall | Method | Vectorization |
|---|---|---|---|---|---|
| Uaememex1 | *0.43* | 0.45 | 0.42 | LR | BERT |
| Uaememex2 | 0.42 | 0.42 | 0.42 | KNN | BERT |
| Uaememex3 | 0.37 | 0.37 | 0.38 | KNN | ASCII |
| Uaememex4 | 0.33 | 0.35 | 0.34 | MLP | D2V |

As we observe, the best result that we obtained was the use of BERT as text vectorization with LR (0.43), while the second (0.42) and third (0.37) best results were obtained using the KNN classifier. On the other hand, we obtained the best results using BERT vectorization, due to its advantage in generating contextualized representations of words, compared to ASCII vectorization, which is a simpler form based on frequencies. However, despite being simple, ASCII vectorization obtained results very close to the more sophisticated BERT vectorization. Its performance was even better than D2V vectorization (0.33).

The parameters we used to obtain the best results are described in Table 3. Regarding text vectorization, we employed the multilingual pre-trained BERT model with a maximum length of 256 features, utilizing the CLS mapping method. In the case of D2V, we tested several parameters; nevertheless, the best result was obtained with a vector size of 100 and a maximum distance between the current and predicted words of 5 (Window). Furthermore, this vector representation was obtained using 40 epochs and word vectors by skip-grams (Dbow words = 1).

**Table 3.**
Vectorization settings and methods

| *Stage* | | *Parameter* | *Values* |
|---|---|---|---|
| Text vectorization | BERT | Model name | BERT-base-multilingual |
| | | Max length | 256 |
| | | Pooling | CLS |
| | D2V | Vector size | 100 |
| | | Window | 5 |
| | | Epochs | 40 |
| | | Dbow words | 1 |
| Classification | LR | Iterations | 500 |
| | | Solver | LBFGS |
| | | Penalty | l2 |
| | MLP | Epochs | 100 |
| | | Loss | Sparse categorical cross entropy |
| | | Activation function | Relu/softmax |
| | | Learning rate | 0.010 |
| | | #parameters | 10303 |
| | KNN | K | 1 |

On the other hand, the LR was iterated 500 times using the Limited-memory BFGS optimization algorithm, which is employed for solving large-scale problems, considering l2 as the criterion penalty. Concerning the KNN algorithm, we tested its performance using different values of $K$ (1, 3, 5, 7, and 9). However, we obtained the best F1 results by using the nearest neighbor *(K=1)*. Regarding MLP, the best model was obtained by 100 epochs, the loss function was Sparse categorical cross entropy, and the learning rate was 0.010. Related to its architecture, we used ReLU activation function in hidden layers, and Softmax in the output layer. Across all layers, the model employed 10303 parameters.

Below, we show in Table 4 the comparison between our team's results and the rest of the competitors. According to the F1-score results, our team, with the best result, obtained the 6th position. Concerning the best team (Ryuan), there is a gap of 0.15. However, it is important to highlight that our experiments were done using straightforward methods, and we only considered OCR texts from the provided dataset.

**Table 4.**
Competition results. The boldface number indicates the best result for our team.

| Team | F1 | Precision | Recall |
|---|---|---|---|
| Ryuan | 0.58 (1) | 0.58 (2) | 0.58 (1) |
| Onarion | 0.57 (2) | 0.58 (1) | 0.56 (2) |
| ymlopez | 0.55 (3) | 0.57 (3) | 0.55 (3) |
| VickBat | 0.52 (4) | 0.54 (4) | 0.51 (4) |
| michaelibrahim | 0.44 (5) | 0.46 (5) | 0.43 (5) |
| Uaememex1 | 0.43 (6) | 0.45 (6) | 0.42 (6) |
| Uaememex2 | 0.42 (7) | 0.42 (7) | 0.42 (6) |
| Uaememex3 | 0.37 (8) | 0.37(8) | 0.38 (7) |
| csuazob | 0.34 (9) | 0.42 (7) | 0.36 (8) |
| Uaememex4 | 0.33 (10) | 0.35 (9) | 0.34 (9) |

Figure 3 shows the confusion matrix of our best result (Uaememex 1), which also evaluates the performance of classification approaches. Based on this matrix, we observe that the majority class (Harmless) remains with the highest number of labeled memes (1307), while the memes labeled as Hate Speech and Inappropriate content included features that could confound memes from other classes. Approximately 520 memes were classified incorrectly, which also represents 22.98% of the complete sample of the training set. This represents a considerable gap in achieving the best possible performance during the training stage, which the proposed approach needs to explore through internal adjustment of parameters from vectorization and classification methods (including LLMs).

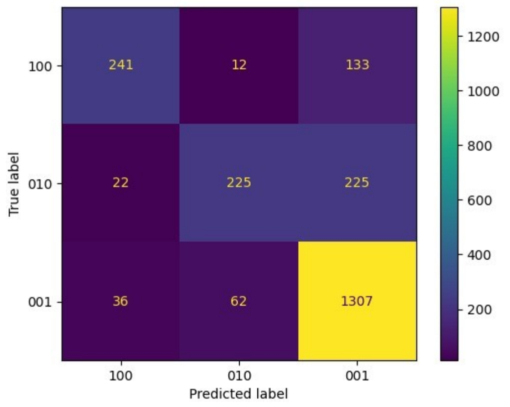

**Figure 3:** Confusion matrix of Uaememex1 in the training stage.

## 5. Conclusions

In this paper, we presented the proposed approach that the Uaememex team employed in Subtask 1 from DIMEMEX-2025, as an effort from the NLP scientific community to minimize digital violence to several communities. In particular, we experimented with lexical and semantic features from different text vectorization methods. We described the stages, parameters, and values to develop classification models for detecting Hate speech, Inappropriate, and Harmless content. Specifically, we employed a framework of classification called NLPClassKit, which focuses on text vectorization methods (OHE, TF-IDF, D2V, BERT, and ASCII) and supervised machine learning algorithms (NB, LR, KNN, MLP, and SVM) to solve NLP classification problems.

According to the experimentation described in Section 4, the configuration that obtained the best results was vectorization of texts through BERT and classification by LR. Despite obtaining lower results than other participants, we noticed that the exploration of semantic features may better contribute to the detection of hate speech, inappropriate content, and harmless content. Nevertheless, we observed that the ASCII-based text representation should be incorporated as a complement to modern classification techniques (see Section 4), as it obtained comparable performance to other approaches.

## Declaration on Generative AI

During the preparation of this work, the authors used Grammarly to check grammar and spelling, as well as Gemini to improve the clarity of some sentences. After using these tools, the authors reviewed and edited the content as needed. Furthermore, they take full responsibility for the content of the publication.

## References

[1] H. Jarquín-Vásquez, I. Tlelo-Coyotecatl, D. I. Hernández-Farías, H. J. Escalante, L. Villaseñor-Pineda, and M. Montes-y-Gómez, "Overview of DIMEMEX at IberLEF2025: Detection of Inappropriate Memes from Mexico," *Proces. del Leng. Nat.*, vol. 75, 2025.

[2] S. M. González-Barba, J. A., Chiruzzo, L., Jiménez-Zafra, "Overview of IberLEF 2025: Natural Language Processing Challenges for Spanish and other Iberian Languages," *Proc. Iber. Lang. Eval. Forum (IberLEF 2025), co-located with 41st Conf. Spanish Soc. Nat. Lang. Process. (SEPLN 2025), CEUR-WS. org*, 2025.

[3] H. Yenala, A. Jhanwar, M. K. Chinnakotla, and J. Goyal, "Deep learning for detecting inappropriate content in text," *Int. J. Data Sci. Anal.*, vol. 6, no. 4, pp. 273–286, 2018, doi: 10.1007/s41060-017-0088-4.

[4] M. Mozafari, R. Farahbakhsh, and N. Crespi, "A BERT-Based Transfer Learning Approach for Hate Speech Detection in Online Social Media," in *Studies in Computational Intelligence*, 2020, vol. 881 SCI, pp. 928–940, doi: 10.1007/978-3-030-36687-2_77.

[5] H. S. Alatawi, A. M. Alhothali, and K. M. Moria, "Detecting White Supremacist Hate Speech Using Domain Specific Word Embedding with Deep Learning and BERT," *IEEE Access*, vol. 9, pp. 106363–106374, 2021, doi: 10.1109/ACCESS.2021.3100435.

[6] J. Rojas-Simón, Y. Ledeneva, and R. A. García-Hernández, "Classification of Human and Machine-Generated Texts Using Lexical Features and Supervised/Unsupervised Machine Learning Algorithms," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 14755 LNCS, pp. 331–341, 2024, doi: 10.1007/978-3-031-62836-8_31.

[7] J. Rojas-Simón, Y. Ledeneva, and A. García-Hernández, "A Dimensionality Reduction Approach for Text Vectorization in Detecting Human and Machine-generated Texts," doi: 10.13053/CyS-28-4-5214.

[8] V. R. Prasetyo and A. H. Samudra, "Hate speech content detection system on Twitter using K-nearest neighbor method," in *AIP Conference Proceedings*, Apr. 2022, vol. 2470, doi: 10.1063/5.0080185.

[9] N. A. Saputra, K. Aeni, and N. M. Saraswati, "Indonesian Hate Speech Text Classification Using Improved K-Nearest Neighbor with TF-IDF-ICSρF," *Sci. J. Informatics*, vol. 11, no. 1, pp. 21–30, 2024, doi: 10.15294/sji.v11i1.48085.

[10] S. Zehra and F. Doja, "A Logistic Regression Model for Hate Speech Recognition," May 2023, doi: 10.4108/eai.24-3-2022.2318769.

[11] A. Pavão *et al.*, "CodaLab Competitions An Open Source Platform to Organize Scientific Challenges," *J. Mach. Learn. Res.*, vol. 24, pp. 1–6, 2023, Accessed: May 20, 2025. [Online]. Available: https://codalab.lisn.fr/.

[12] J. Rojas-Simon, Y. Ledeneva, and R. A. Garcia-Hernandez, "Evaluation of Text Summaries Based on Linear Optimization of Content Metrics," vol. 1048, p. 215, 2022, doi: 10.1007/978-3-031-07214-7.

[13] Q. Le and T. Mikolov, "Distributed Representations of Sentences and Documents."

[14] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," pp. 4171–4186, Accessed: May 20, 2025. [Online]. Available: https://github.com/tensorflow/tensor2tensor.