ISSN 1870-4069

Online Sexism Detection Models

Karen Esmeralda Delgado Pérez, Néstor David García Franco,
Darnes Vilariño Ayala, Beatriz Martínez Beltrán,
David Eduardo Pinto Avendaño

Benemérita Universidad Autónoma de Puebla,
Mexico

{dp224470254, gf224470255@alm.buap.mx},
{darnes.vilarino, beatriz.beltran, david.pinto}@correo.buap.mx

Abstract. The automatic detection of online sexism represents a key challenge for content analysis in social networks, as current binary classification models are not always able to address all that sexist content may present. The present study focuses on Task 10: Task A of SemEval 2023, which consists of classifying comments as sexist or non-sexist. Two machine learning models based on natural language processing (NLP) techniques are presented. Unlike previous work focused on sentiment analysis, this approach is explicitly approached with the definition of sexism adopted by SemEval. The evaluation, performed on a dataset of Reddit and Gab comments, demonstrates that unigram-based models outperform bigram and trigram-based models in classification accuracy. This work seeks to advance the accurate and explainable detection of sexism in digital environments.

Keywords: Sexism, automatic learning, NLP.

1 Introduction

In a world where technology is constantly advancing, sexism has evolved, finding new forms of expression, such as online harassment and content that promotes gender-based hate. Despite efforts to achieve gender equality, there are still stereotypes and barriers that limit access to equal opportunities. From the wage gap to unequal representation in leadership positions, sexism affects both women and men, although it impacts differently and, in many cases, mainly women.

Sexism is understood as the discrimination of a person based on gender, promoting the idea that one sex is superior to the other and based on pre-established beliefs for one gender, negatively impacting the people affected. It is expressed in the use of language, with prejudicial attitudes about gender roles and practices that hinder access to equal rights and opportunities. Online sexism is a form of harassment, primarily against women, that aggressively focuses on minimizing their progress, making them feel assaulted and humiliated.

Automated tools help detect sexism on a large scale. However, binary detection ignores the diversity of sexist content that exists and does not provide a clear explanation of why something is sexist, generating distrust and reducing its effectiveness. To address this issue, SemEval Task 10: SemEval 2023 Task Explainable Detection of Online Sexism (EDOS) is presented [1].

Task 10 is derived from shared tasks that dealt with the detection of abuse and hate. This task proposes and applies a taxonomy with 3 tasks: Task A is in charge of detecting whether the content is sexist or not; Task B classifies to which category of sexism it belongs and Task C subcategorizes the comments [1].

This paper addresses Task A of SemEval Task 10: detecting whether or not content is sexist with 2 different models, which determine whether comments are positive, negative or neutral, by using natural language processing techniques such as tokenization and lemmatization. The results are also compared with RoBERTa, a pre-trained model for content evaluation.

This article is organized to contextualize the results of the models that participated in SemEval 2023(Section 2). Section 3 presents the methodology of the implemented models, including the integration of LSA and LDA, meanwhile Section4 evaluates the effectiveness of models. Section 5 ends with the main conclusions of the study and future work.

2 Related Work

Among the top-performing systems presented at SemEval 2023, multiple models or ensemble-based approaches were used. Many of them applied additional training to their models and multitask learning. The teams that stood out the most were *stce* and *PASSTeam*, both of which used multitask learning and additional pre-training, obtaining better results on two or more tasks. [1].

The team *stce* used RoBERTa-large [2] and ELECTRA [3]. while the team *PASSTeam* used a multitask learning strategy [4] with tuned versions of RoBERTa y HateBERT [5].

For Task A, *PingAnLifeInsurance* team used a multitask neural network framework [6], in addition to performing additional pre-training with DeBERTa-v3 [7] and TwHIN-BERT [8] using unlabeled data and an additional Kaggle dataset. On the other hand, *FiRC-NLP* occupied a set of DeBERTa models fitted exclusively with the labeled task data.

In the case of Task B, *JUAGE* stood out as one of the few systems that used (prompt-based learning), achieving first place using the PaLM model tuned with [9] instructions for parameter optimization and majority voting over six iterations.

Moreover, the *PALI* team performed additional pre-training of DeBERTa-v3 with unlabeled data and included a second loss term based on the scaled cross-entropy, to address Task C.

Overall, Task B and Task C scores showed below average results and greater variability compared to Task A. All participating systems outperformed the simpler baseline, which was to predict the most frequent class, and most also outperformed a more complex baseline base on DeBERTa-v3 with continuous pre-training.

3 Proposed Methodology

The methodology proposed for the development of this research, which can be seen in Algorithm 1, is shown below 1.

Algorithm 1 Sentiment Analysis Process

Require: Complete Corpus C with n documents
Ensure: Classified polarity and generated vocabulary
1: $C_{limpio} \Leftarrow$ Formating(C) ▷ Cleans and saves each document
2: $Training$, $Test \Leftarrow$ RandomSampling(C_{limpio})
3: **for all** document d en $Training$ **do**
4: $polarity_d \Leftarrow$ AnalyzeFeelings(d) ▷ Classifies: positive, negative o neutral
5: $Vocabulary \Leftarrow Vocabulary \cup$ ExtractWords(d)
6: **end for**
7: **return** Polarity, Vocabulary

3.1 Format

It is in charge of cleaning the corpus data. It starts by reading the corpus, then processes the data of each document, filtering its columns depending on whether they contain information or not, removes common or repetitive words(stopwords) from the comments, and creates a folder that stores text files generated by each processed row and saves each of the documents in a new file organized by row.

3.2 Sampling

Subsequently, the set of files is randomly organized into two groups: one for training and one for test. Eighty percent of the documents are assigned for training and the remaining 20% for testing. To do this, we go through the subfolders generated in the previous point, selecting the files to be processed. It calculates how many files should be in the training set and randomly select the documents.In addition, two new subfolders are created, one for training data and one for testing. Finally, the selected files for the training and test sets are copied to the corresponding training and test folders.

3.3 Sentiment Analyzer

The VADER sentiment analyzer, which assigns a polarity score(positive, negative, neutral) to the content, and the Google Translate API were used to classify the sentiment of the comments.

The training set files are used to verify that the text is valid and that the file is not empty. Then, to classify the opinion, the text found in each of the files is translated into the following text.If the translation fails because the comment does not exist or the file is empty, the empty, the sentiment is considered neutral. Otherwise, advance to the VADER analyzer to obtain the polarity score of the translated text. If the score is greater than 0, it is considered a positive comment; if the score is equal to 0, it is taken as neutral, and it is negative when the score is less than 0.

3.4 Vocabulary

Each text file is processed to extract the words and generate a vocabulary. Each word contained is related to the files where it appears.

To do this, we went through each of the subfolders in the indicated directory, read the text in each of the previously generated files, modified the text to lowercase, divided it by words, and returned to a file that includes all the words collected.

3.5 LSA

Latent Semantic Analysis(LSA) is a technique for creating vector representations of texts that aim to capture tptheir semantic content. The main function of LSA is to calculate the similarity of pairs of texts by comparing their vector representations. [11].

Before working with LSA a preprocessing of the content was performed, which includes cleaning up text (removing URLS, numbers, non-alphabetic characters and reducing spaces), separating the text into tokens, removing common words(in this particular case we included additional stopwords with a total of 1298 words [12], plus stopwords included by Spacy [23]) and lemmatize. The corpus cleans its save.

Subsequently, TF-IDF matrices were generated, used to calculate the relevance of a word within a document, for different n-grams (unigram, bigram, trigram). An n-gram is understood as a set of n consecutive elements for a text document [15].

3.6 LDA

Latent Dirichlet allocation (LDA) is a generative probabilistic model for collections of discrete data, such as text corpus. LDA is a three-level hierarchical Bayesian model in which each element of a collection is modeled as a finite mixture over an underlying set of topics. In turn, each topic is modeled as an infinite mixture over an underlying set of topic probabilities [13].

4 Results Analysis

In this study, different algorithms designed in Python were applied for Task A: detecting whether the content is sexist or not, classifying it as positive, negative, or neutral. Two proposals are presented: the first one consists of the Formatter, Sampler, Sentiment Analyzer and Vocabulary. The second proposal consists of the application of the LDA and LSA algorithms.

As mentioned above, the methodology considers two proposals. The first one, consists in the consideration of n-grams, applied both to the training set and to the test set.

The second proposal applies LSA using TruncatedSVD from sklearn [10], to reduce the dimensionality of the feature space to a number of principal components and continued with a pretrained RoBERTa sentiment analyzer, the results are categorized as positive, negative, and neutral, and saved along with their polarity. This is done for both training and set.

A Hugging Face [24] feature was also added, which creates a specialized sentiment analysis pipeline. A pipeline is a series of preconfigured steps that allows specific tasks (such as sentiment analysis, translation, etc.) to be performed without having to configure each part of the process manually. In this case, the pipeline is configured to evaluate whether a text is positive, negative, or neutral. Finally, a graph of the sentiment distribution is generated.

After applying LSA, the text data were loaded and preprocessed. The TfidfVectorizer library is used to transform the texts into numerical feature vectors; converting the text into a numerical representation in which each word is represented by its TFIDF, which allows to measure the relative importance of each word in the corpus.

Class balancing techniques (SMOTE [20]) were applied, which helps to improve the performance of the models. By applying PCA (Principal Component Analysis) they were reduced to 2,000 principal components.

In addition, it was evaluated with two classification models (LDA and Random Forest) to predict the sentiment of the texts. Finally, confusion matrices were obtained, showing how many instances were correctly and incorrectly classified, and classification reports for both models (precision, recall, F1-score), which allowed us to evaluate the performance of the models.

4.1 Dimension of the Dataset

These models were trained from a corpus with one million unlabeled Reddit comments provided by SemEval (2023). Included within the dataset are uppercase, numeric, emoji, urls, and ASCII characters.

4.2 Evaluation of the Models

Training is used to build the model and it allows the classification of the test set. The dataset of 962,122 for the training set consists of the 769,696 documents, and the test set consists of 192,425 documents. To make the LSA and LDA tests more accurate, we used different n-grams (unigrams, bigrams and trigrams).

Where Accuracy shows the accuracy of the model; Precision is defined as how close we are to the data to be obtained; Recall is described as the ratio of true positive to false positive plus false negative; F1-Score is a harmonic measure

Table 1. Results obtained for unigrams with LDA.

$$\begin{bmatrix} 57531 & 49603 & 4111 \\ 9242 & 94521 & 7431 \\ 2166 & 38825 & 70248 \end{bmatrix}$$

	Unigram			
N-grams	Precision	Recall	F1-Score	Support
Negative	0.83	0.52	0.64	111245
Neutral	0.52	0.85	0.64	111194
Positive	0.86	0.63	0.73	111239
Accuracy			0.67	333678
Macro avg	0.74	0.67	0.67	333678
Weighted avg	0.74	0.67	0.67	333678

Table 2. Results obtained for unigrams with Random Forest.

$$\begin{bmatrix} 84694 & 22600 & 3951 \\ 13571 & 92338 & 5285 \\ 1359 & 6426 & 103454 \end{bmatrix}$$

	Unigram			
N-grams	Precision	Recall	F1-Score	Support
Negative	0.85	0.76	0.80	111245
Neutral	0.76	0.83	0.79	111194
Positive	0.92	0.93	0.92	111239
Accuracy			0.67	333678
Macro avg	0.84	0.84	0.84	333678
Weighted avg	0.84	0.84	0.84	333678

between precision and recall; Support is to increase the number of rare cases instead of simply duplicating the existing cases.

When evaluating the model, it is observed that the use of unigrams results in better accuracy compared to the results obtained using bigrams and trigrams.

The precision of the algorithm with unigrams with LDA is given for negative 0.83; neutral 0.52 and positive 0.86. See Table 1.

In the case of unigrams with Random Forest, negative 0.85; neutral 0.76 and positive 0.92 are obtained. See Table 2.

The classification precision for LDA bigrams was 0.72 for negative; 0.36 for neutral and 0.68 for positive. See Table 3.

In the case of bigrams with Random Forest it was 0.76 for negative; 0.38 neutral and 0.77 positive. See Table 4.

The results obtained for trigrams with LDA were negative 0.64; neutral 0.34 and positive 0.57. See Table 5.

Table 3. Results obtained for bigrams with LDA

$$\begin{bmatrix} 14009 & 94363 & 2873 \\ 3129 & 104819 & 3246 \\ 2235 & 95766 & 13238 \end{bmatrix}$$

	Bigram			
N-grams	Precision	Recall	F1-Score	Support
Negative	0.72	0.13	0.21	111245
Neutral	0.36	0.94	0.52	111194
Positive	0.68	0.12	0.20	111239
Accuracy			0.40	333678
Macro avg	0.59	0.40	0.31	333678
Weighted avg	0.59	0.40	0.31	333678

Table 4. Results obtained for bigrams with Random Forest

$$\begin{bmatrix} 25874 & 82152 & 3219 \\ 6367 & 100749 & 4078 \\ 1835 & 85335 & 24069 \end{bmatrix}$$

	Bigram			
N-grams	Precision	Recall	F1-Score	Support
Negative	0.76	0.23	0.36	111245
Neutral	0.38	0.91	0.53	111194
Positive	0.77	0.22	0.34	111239
Accuracy			0.45	333678
Macro avg	0.63	0.45	0.41	333678
Weighted avg	0.63	0.45	0.41	333678

While the precision in trigrams with Random Forest was negative 0.78; neutral 0.34 and positive 0.84. See Table 6.

To complement and contrast the results obtained using the LSA and LDA methodology, we incorporated the RoBERTa model, which has demonstrated outstanding results in text classification task, and is used for large volumes of unstructured data. In order to use RoBERTa, we applied a binary labeling scheme (sexism/no-sexism) which allowed us to evaluate its effectiveness against n-gram-based methods. This model improves the analysis of the entire data set (962,122). Show the results in the table 7.

The results obtained in the sexism class are based on the semantic and contextual complexity of the language. Unlike explicit non-sexist comments, the expressions are often composed in a subtle, ironic, or implicit manner, which makes them difficult to detect.

Table 5. Results obtained for trigrams with LDA.

$$\begin{bmatrix} 1634 & 108939 & 672 \\ 530 & 110174 & 490 \\ 370 & 109309 & 1560 \end{bmatrix}$$

	Trigram			
N-grams	Precision	Recall	F1-Score	Support
Negative	0.64	0.01	0.03	111245
Neutral	0.34	0.99	0.50	111194
Positive	0.57	0.01	0.03	111239
Accuracy			0.34	333678
Macro avg	0.52	0.34	0.19	333678
Weighted avg	0.52	0.34	0.19	333678

Table 6. Results obtained for trigrams with Random Forest.

$$\begin{bmatrix} 2904 & 108081 & 260 \\ 676 & 110310 & 208 \\ 150 & 108653 & 2436 \end{bmatrix}$$

	Trigram			
N-grams	Precision	Recall	F1-Score	Support
Negative	0.78	0.03	0.05	111245
Neutral	0.34	0.99	0.50	111194
Positive	0.84	0.02	0.04	111239
Accuracy			0.35	333678
Macro avg	0.65	0.35	0.20	333678
Weighted avg	0.65	0.35	0.20	333678

5 Conclusions and Future work

Online sexism has represented an important challenge, where anonymity and accessibility to platforms have facilitated the propagation of hate speech and gender discrimination. Throughout this study, various strategies for automatic detection of sexism have been analyzed, highlighting the effectiveness of models based on transformers and deep learning approaches. Previous results reflect that models such as RoBERTa, DeBERTa-v3 and HateBERT can identify patterns of sexism with high accuracy, although challenges remain in classifying ambiguous cases and explaining the detected biases.

The analysis of n-grams and techniques such as Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA) have provided a better understanding of the linguistic structure of sexist content, providing a more detailed approach to detection. However, there is a need for further development

Table 7. Results with RoBERTa.

	RoBERTa			
Etiquetas	Precision	Recall	F1-Score	Support
Sexism	0.0357	0.0357	0.0357	36498
No-sexism	0.9620	0.9620	0.9620	925622
Accuracy			0.92689	962120
Macro avg	0.4989	0.4989	0.4989	962120
Weighted avg	0.9268	0.9268	0.9268	962120

of models that not only classify content but also provide clear explanations as to why a given text is considered sexist.

However, it is necessary to continue to develop models that not only classify content but also provide clear explanations as to why a given text is considered sexist. As future work, we plan to improve the models presented, extend the study to tasks B and C of SemEval, in order to classify the different categories and subcategories of the sexism; to analyze common errors in the classifications in order to refine both preprocessing and representation of the text, and to incorporate refine both the preprocessing and the representation of the text, and incorporate other pretrained other pretrained models, such as DeBERTa and HateBERT for sexism detection.

References

1. Kirk, Hannah, Wenjie Yin, Bertie Vidgen, and Paul Röttger. "SemEval-2023 Task 10: Explainable Detection of Online Sexism." In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, edited by Atul Kr. Ojha, A. Seza Doğruöz, Giovanni Da San Martino, Harish Tayyar Madabushi, Ritesh Kumar, and Elisa Sartori, 2193–2210. Toronto, Canada: Association for Computational Linguistics, 2023. https://aclanthology.org/2023.semeval-1.305/.doi:10.18653/v1/2023.semeval-1.305.
2. Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. "RoBERTa: A Robustly Optimized BERT Pretraining Approach." ArXiv preprint arXiv:1907.11692, 2019. https://arxiv.org/abs/1907.11692.
3. Clark, Kevin, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators." In *International Conference on Learning Representations (ICLR)*, 2020. https://openreview.net/forum?id=r1xMH1BtvB.
4. Yu, Tianhe, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. "Gradient Surgery for Multi-Task Learning." In *Advances in Neural Information Processing Systems*, volume 33, 5824–5836. Curran Associates, Inc., 2020. https://proceedings.neurips.cc/paper/2020/hash/3fe230348e9a12c13120749e3f9fa4cd-Abstract.html.
5. Caselli, Tommaso, Valerio Basile, Jelena Mitrović, and Michael Granitzer. "HateBERT: Retraining BERT for Abusive Language Detection in English."

In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, 17–25. Online: Association for Computational Linguistics, 2021. https://aclanthology.org/2021.woah-1.3/.

6. Liu, Xiaodong, Yu Wang, Jianshu Ji, Hao Cheng, Xueyun Zhu, Emmanuel Awa, Pengcheng He, Weizhu Chen, Hoifung Poon, Guihong Cao, and Jianfeng Gao. "The Microsoft Toolkit of Multi-Task Deep Neural Networks for Natural Language Understanding." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 118–126. Online: Association for Computational Linguistics, 2020. https://aclanthology.org/2020.acl-demos.14/.
7. He, Pengcheng, Jianfeng Gao, and Weizhu Chen. "DeBERTaV3: Improving DeBERTa Using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing." ArXiv preprint arXiv:2111.09543, 2021. https://arxiv.org/abs/2111.09543.
8. Zhang, Xinyang, Yury Malkov, Omar Florez, Serim Park, Brian McWilliams, Jiawei Han, and Ahmed El-Kishky. "TwHIN-BERT: A Socially-Enriched Pre-trained Language Model for Multilingual Tweet Representations." ArXiv preprint arXiv:2209.07562, 2022. https://arxiv.org/abs/2209.07562.
9. Chowdhery, Aakanksha, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. "PaLM: Scaling Language Modeling with Pathways." ArXiv preprint arXiv:2204.02311, 2022. https://arxiv.org/abs/2204.02311.
10. Scikit-learn. (2025). *Installation Guide*. Recuperado de https://scikit-learn.org/stable/install.html
11. P. Wiemer-Hastings, K. Wiemer-Hastings, and A. Graesser. (2004, November). *Latent semantic analysis*. En *Proceedings of the 16th international joint conference on Artificial intelligence*, pp. 1-14.
12. Stopwords ISO. (n.d.). Recuperado de https://github.com/stopwords-iso/stopwords-en/blob/master/stopwords-en.txt
13. D. M. Blei, A. Y. Ng, and M. I. Jordan. (2003). *Latent dirichlet allocation*. Journal of Machine Learning Research, 3(Jan), 993-1022.
14. Zahraa Berjawi. (2022). *Benevolent Sexism Detection in Text: A Data-Centric Approach*. PhD Thesis.
15. MathWorks. (n.d.). *N-grams*. Recuperado de https://la.mathworks.com/discovery/ngram.html.
16. G. T. Rahutami and F. Z. Ruskanda. (2023). *Sexism Detection and Classification Using RoBERTa and Data Augmentation*. En *2023 10th International Conference on Advanced Informatics: Concept, Theory and Application (ICAICTA)*, Lombok,

Indonesia, pp. 1-6. doi: 10.1109/ICAICTA59291.2023.10390414. keywords: Training;Social networking (online);Data augmentation;Transformers;Data models;Informatics;sexism;text classification;RoBERTa;data augmentation

17. H. Mohammadi, A. Giachanou, and A. Bagheri. (2024). *A transparent pipeline for identifying sexism in social media: Combining explainability with model prediction*. Applied Sciences, 14(19), 8620.
18. E. Martinez, J. Cuadrado, J. C. Martinez-Santos, and E. Puertas. (2023). *Detection of Online Sexism Using Lexical Features and Transformer*. En *2023 IEEE Colombian Caribbean Conference (C3)*, Barranquilla, Colombia, pp. 1-5. doi: 10.1109/C358072.2023.10436298. keywords: Social networking (online);Merging;Linguistics;Transformers;Feature extraction;Natural language processing;Global communication;Transformers;Lexical Features;Social Media;Misogyny;Sexism
19. A. Chaudhary and R. Kumar. (2023). *Sexism Identification In Social Networks*. En *CLEF (Working Notes)*, pp. 891-900.
20. Imbalanced-learn. (2025). *SMOTE*. Recuperado de `https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html`.
21. Scikit-learn. (2025). *sklearn metrics — Metrics and scoring*. Recuperado de `https://scikit-learn.org/stable/api/sklearn.metrics.html`.
22. Bird, Steven, Ewan Klein, and Edward Loper. (2009). *Natural Language Processing with Python*. O'Reilly Media. Recuperado de `https://books.google.com/books/about/Natural_Language_Processing_with_Python.html?hl=es&id=Au-_DwAAQBAJ`.
23. spaCy. (2025). *spaCy 101: Everything you need to know*. Recuperado de `https://spacy.io/usage/spacy-101`.
24. Hugging Face. (2025). *Models – Hugging Face*. Recuperado de `https://huggingface.co/models`.