ANLP-Uniso at EXIST 2025: Sexism Identification and Characterization in Tweets

Notebook for the sEXism Identification in Social neTworks Lab at CLEF 2025

Ghada Ben Amor[1,*], Nawres Medimagh[1], Saoussen Ben Chaabene[1] and Omar Trigui[1,2]

[1]*University of Sousse, The Higher Institute of Management of Sousse, Sousse 4000, Tunisia*

[2]*Miracl laboratory, University of Sfax, Tunisia*

Abstract

The rise of sexist discourse on social media platforms, especially Twitter, has become a pressing societal concern, necessitating the development of automated detection systems. In this study, we present a comprehensive approach to detecting sexist content in Spanish tweets as part of the EXIST 2025 shared task at CLEF. Leveraging the multilingual T5 (mT5) model for contextual embeddings, our system integrates a variety of machine learning and deep learning classifiers, including traditional machine learning approaches (Logistic Regression and SVM) and neural networks (RNN, GRU, and hybrid FNN+GRU). To enhance classification accuracy, we apply extensive preprocessing, feature normalization, dimensionality reduction via PCA, and data balancing techniques such as SMOTE and class weighting. Our experiments show that while simpler models like Logistic Regression achieve strong performance, ensemble strategies further improve robustness. The results underscore the value of combining transformer-based embeddings with classical and neural classifiers to address the nuanced challenge of online sexism detection.

Keywords

Sexism detection, Social media, Twitter, Spanish tweets, EXIST 2025, CLEF, Machine learning, Deep learning

1. Introduction

The rapid proliferation of social media platforms has transformed the way individuals communicate and express opinions. However, this digital revolution has also given rise to significant challenges, particularly in the form of online abuse and discrimination. Among these, sexist content ranging from explicit harassment to subtle gender-based bias has become alarmingly prevalent, especially on platforms like Twitter. The automatic detection of such harmful language is thus a crucial task in the broader effort to foster safer and more inclusive online environments.

Traditional natural language processing (NLP) techniques have struggled to accurately identify sexism due to its nuanced and context-dependent nature. Modern approaches, however, increasingly rely on machine learning (ML) and deep learning (DL) methods that can capture more complex linguistic patterns. In particular, transformer-based models such as mT5, which leverage large-scale pretraining and contextual embeddings, have demonstrated significant advances in understanding and processing human language. This study aims to develop an effective sexism detection system by combining state-of-the-art transformer models with traditional ML/DL classifiers, enhanced through robust preprocessing, feature extraction, and handling of class imbalance.

By applying this hybrid approach to social media data, we seek not only to improve detection accuracy but also to gain deeper insights into the linguisticcharacteristics of sexist discourse online.

With the exponential rise of social media platforms, especially Twitter, sexist discourse has found new ways to spread rapidly and widely. Whether explicit or implicit, such content significantly impacts gender perception and reinforces stereotypes and online abuse.

CLEF 2025 Working Notes, 9 – 12 September 2025, Madrid, Spain

*Corresponding author.
benamorghada@isgs.u-sousse.tn (G. B. Amor); medimaghnaoures@isgs.u-sousse.tn (N. Medimagh); benchaabensaoussen@isgs.u-sousse.tn (S. B. Chaabene); omar.trigui@isgs.u-sousse.tn (O. Trigui)

Automatically detecting this type of language has thus become a critical task. However, it remains particularly challenging due to the subjective nature of language, cultural and linguistic diversity, and the often-subtle manifestations of sexism. Additionally, the class imbalance in the dataset where sexist tweets are fewer than non-sexist ones further complicate the development of effective machine learning systems. The central research question of this study is: How can we develop a reliable, robust, and multilingual system for the automatic detection of sexist content on social media particularly Spanish-language tweets while ensuring both interpretability and generalization?

The main objective of this study is to design and implement an automatic sexism detection system for tweets, as part of the EXIST 2025 shared task at the CLEF campaign. To achieve this, we rely on:

- Contextual text representations generated using the mT5 multilingual transformer model.
- A combination of traditional machine learning models (such as Logistic Regression and SVM) and deep learning architectures (such as RNN, GRU, FNN, and hybrid models).
- Rigorous text preprocessing techniques to ensure the linguistic consistency and quality of the dataset.
- Techniques for dimensionality reduction (PCA), data balancing (SMOTE, class weighting), and robust evaluation (F1-score, confusion matrices, accuracy, precision, macro avg, weighted avg, support).

2. Related Work

The automatic detection of online sexism has become a growing area of research within Natural Language Processing, particularly due to the proliferation of gender-based hate speech on social media platforms. Since the launch of the EXIST shared task in 2021, the community has developed diverse methodologies for handling the binary and fine-grained classification of sexist content in multiple languages. Early approaches to sexism detection primarily relied on classical machine learning algorithms such as logistic regression, support vector machines [SVMs], and random forests using hand-crafted features like TF-IDF [**1**]. Although these models provided lightweight and interpretable solutions, their performance often lagged behind neural approaches, especially in capturing implicit and contextual sexism. The widespread adoption of transformer-based architectures, such as BERT, mBERT, and XLMRoBERTa, has significantly improved the performance of sexism detection systems [**2**]. These models leverage contextual embeddings to better understand the subtle and nuanced nature of online discourse. Khan et al. [**3**] addressed multilingual sexism detection by leveraging transformer models like XLM-RoBERTa and mBERT. They proposed an ensemble approach combining multiple fine-tuned models to identify explicit and implicit sexism in both English and Spanish texts, achieving excellent performance in the EXIST 2024 shared task. The SemEval-2023 Task 10 further expanded the classification framework by introducing a hierarchical taxonomy of sexism, distinguishing between threats, stereotypes, and derogatory remarks. These systems often combine fine-tuned transformer models (e.g., DeBERTa-v3, TwHIN-BERT) with auxiliary techniques like multi-label learning and contextual data augmentation [**4**]. Hybrid approaches have also been proposed to integrate textual and non-textual signals. A notable example is the use of ByT5 a byte-level multilingual transformer combined with TabNet for incorporating structured metadata such as platform, language, and readability [**5**]. While promising, such systems remain computationally expensive and require extensive fine-tuning to outperform simpler baselines. Finally, Azadi et al, tackled bilingual sexism detection using fine-tuned XLM-RoBERTa and GPT-3.5 few-shot learning. Their approach showed high performance in English and Spanish, demonstrating the effectiveness of both fine-tuning and prompt-based methods for identifying sexism with minimal annotated data [**6**].

3. Methodology

3.1. Dataset Description: EXIST 2025 Tweets Dataset

As part of our research on automatic sexism detection in social media content, we utilized the EXIST 2025 Tweets Dataset, released by the organizers of the CLEF 2025 conference under the shared task entitled Explainable Detection of Sexism in Social Networks. This multilingual dataset focuses primarily on tweets written in Spanish and English, with the goal of enabling the development of explainable AI models capable of identifying and classifying sexist content. The dataset is organized into three main subdirectories.

3.2. Text Preprocessing

The preprocessing phase was essential to ensure the quality and linguistic consistency of the dataset prior to model training. Since the EXIST 2025 dataset contains multilingual content, including both English and Spanish tweets, we applied a language filtering step to retain only tweets written in Spanish, in line with the objectives of our study. To prepare the text data, we first performed a normalization process that involved the removal of irrelevant textual elements such as hyperlinks, user mentions, and hashtags. This was followed by the conversion of emojis into their textual representations to preserve semantic information, which were subsequently cleaned to avoid introducing noise. Next, we eliminated all special characters and non-alphanumeric symbols, except for characters specific to the Spanish language, such as accented vowels and the letters ñ and ü. This step helped maintain the integrity of Spanish orthography while reducing unnecessary variability in the data. We also removed common Spanish stopwords to reduce redundancy and focus on the most informative components of the tweets. Finally, any tweets that could not be reliably identified as Spanish or became empty after preprocessing were excluded from the dataset. This cleaning process resulted in a more homogeneous and relevant dataset, optimized for the detection of sexist content in Spanish-language tweets.

3.3. Libraries and Tools

This study employed a range of Python libraries to handle data preprocessing, feature extraction, model training, and evaluation for sexism detection in tweets.

- **Emoji**
  The emoji library was used to process emojis, which are common in social media texts and can carry semantic or emotional information [7]. Emojis were either removed or analyzed as features, as certain emojis may correlate with sexist language.

- **CatBoost, LightGBM, and XGBoost**
  These gradient boosting libraries were used to train classification models [8]. CatBoost is particularly efficient with categorical features, while LightGBM and XGBoost offer fast, scalable implementations. All were evaluated for their performance in binary classification and feature importance analysis.

- **Transformers (mT5)**
  The transformers library from Hugging Face provided access to the pre-trained mT5 model, which was used to generate contextualized text embeddings [5]. These embeddings capture deep semantic meaning and are well-suited for nuanced NLP tasks like sexism detection.

- **Scikit-learn (sklearn)**
  Used for essential machine learning tasks including data splitting, preprocessing, baseline modeling, and evaluation through metrics such as accuracy and F1-score [9].

- **TensorFlow, PyTorch, and SciKeras**
  These libraries supported the development and evaluation of deep learning models. SciKeras enables integration of Keras models within scikit-learn pipelines, combining deep learning with traditional ML workflows [10].

- **Nltk**

Provided basic NLP functions such as tokenization, stop word removal, and lemmatization to prepare tweets for modeling [**11**].

- **Langdetect**
  Automatically identified the language of each tweet, ensuring that only Spanish language content was processed [**12**].

- **Imbalanced-learn (imblearn)**
  Addressed class imbalance using techniques such as SMOTE to improve the model's ability to detect minority (sexist) classes [**13**].

- **Matplotlib and seaborn**
  These visualization libraries were used for exploratory data analysis and to graphically present classification results and feature distributions [**14**].

3.4. Applied Techniques

- **Text Embedding Using mT5 Encoder**
  We used the multilingual T5 (mT5) model from Google to generate contextual embeddings of the tweets. The model extracts dense vector representations from the text using the encoder part of mT5, allowing the input data to be transformed into meaningful numerical features suitable for machine learning models. Each embedding corresponds to a sentence-level representation averaged over the token dimension.

- **Feature Normalization**
  Before dimensionality reduction, the extracted embeddings were normalized using z-score normalization (Standard scaling). This technique ensures that each feature has a mean of zero and a standard deviation of one, which is essential for models sensitive to scale, such as PCA.

- **Dimensionality Reduction Using Principal Component Analysis (PCA)**
  PCA was applied to reduce the dimensionality of the embedding vectors from their original size (e.g. 768 dimensions for mt5-base) to 50 principal components. This reduces computational Front matter complexity, alleviates noise, and allows for better visualization and training efficiency without significant loss of information.
- **Label Encoding**
  Categorical labels (e.g., "sexist", "non-sexist") were converted into numeric format using label encoding. This step is necessary for machine learning models that require numerical input for training.

- **Data Balancing Using SMOTE**
  The Synthetic Minority Over-sampling Technique (SMOTE) was applied to the training data to address class imbalance. It generates synthetic examples of the minority class by interpolating between existing minority class samples, improving model generalization and reducing bias toward the majority class.

- **Class Weighting**
  In addition to SMOTE, class weighting was used to handle imbalance by assigning higher weights to the minority class during model training. This encourages the classifier to pay more

attention to underrepresented samples, improving performance on the minority class without altering the dataset itself.

- **Cosine Similarity Analysis**
  A cosine similarity matrix was generated using the final training embeddings. This matrix measures the pairwise similarity between tweets in the embedding space, useful for exploratory data analysis or understanding the semantic relationships between samples.

- **Classification Models**
  To perform the classification task, a diverse set of models was trained and evaluated. Among the traditional machine learning models, we employed Logistic Regression, Support Vector Machines (SVM), Random Forest, Naive Bayes, k-Nearest Neighbors (KNN), LightGBM, XGBoost, and CatBoost. These models were trained using PCA reduced embeddings, and class weights were applied to address the imbalance in the dataset. In addition to these, neural network architectures were also explored. We implemented Feedforward Neural Networks (FNN), which consist of multiple dense layers with dropout regularization, as well as recurrent models such as RNNs and GRUs, capable of capturing sequential dependencies in the input data. Furthermore, we tested hybrid architectures combining FNN with GRU and RNN with GRU to assess potential synergies between feedforward and recurrent mechanisms.

- **Evaluation Metrics and Confusion Matrices**
  Models were evaluated using standard classification metrics: accuracy, precision, recall, and F1-score. Confusion matrices were visualized to better understand each model's performance on the validation set.

- **Ensemble Prediction with Fallback Mechanism**
  Final predictions on the test set were made using a majority voting strategy across all trained models. In case all predictions failed for a sample, a fallback to a subset of more robust models was applied. This ensemble approach improves robustness and leverages the diversity of models.

4. Experiments and Results

4.1. Logistic Regression

Logistic Regression achieved the highest performance among all models, with an accuracy of 0.618, precision of 0.619, recall of 0.618, and F1-score of 0.617. The confusion matrix shows that the model correctly predicted 176 "NO" instances and 152 "YES" instances, while misclassifying 89 "NO" and 114 "YES" cases. The balanced precision and recall values indicate that the model generalizes well without significant bias toward either class. This suggests that Logistic Regression is robust for this dataset, likely due to its ability to handle linear decision boundaries effectively.

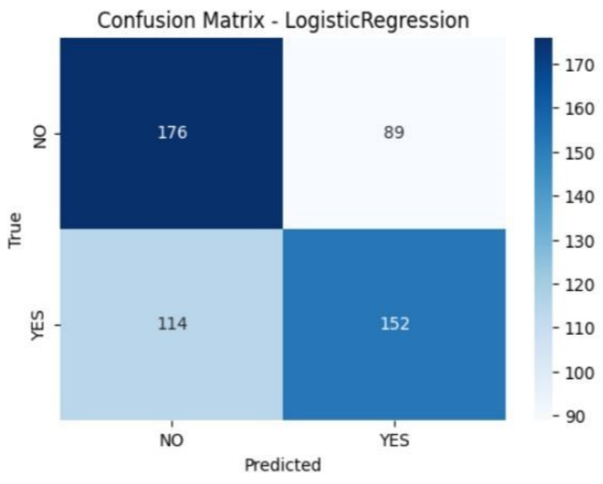


Figure 1: Confusion matrix for Logistic Regression

Figure1 illustrates the confusion matrix for Logistic Regression, while Table 1 presents the corresponding classification report.

Table 1
Classification report for Logistic Regression

Class	Precision	Recall	F1-score	Support
NO	0.61	0.66	0.63	265
YES	0.63	0.57	0.60	266
Accuracy	–	–	0.62	531
Macro avg	0.62	0.62	0.62	531
Weighted avg	0.62	0.62	0.62	531

4.2. SVM

The SVM model achieved an accuracy of 0.595 and an F1-score of 0.594. While the performance is comparable to other models, it does not stand out. The similar precision and recall values indicate balanced classification, but the overall metrics suggest that the kernel or parameters used may not be optimal for this dataset. Experimentation with different kernels or regularization parameters could yield better results.

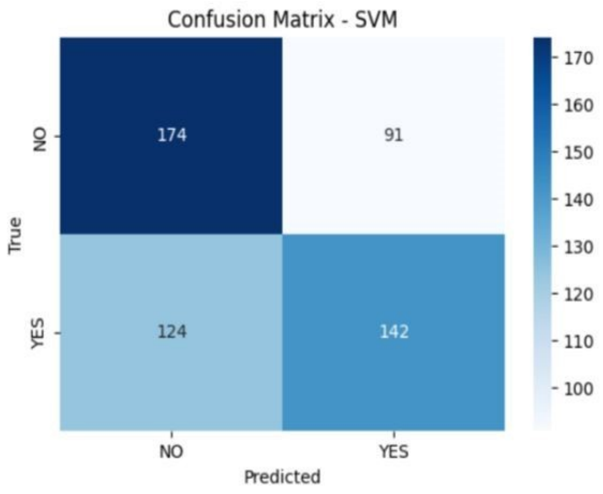


Figure 2: Confusion matrix for SVM

Figure 2 illustrates the confusion matrix for SVM, while Table 2 presents the corresponding classification report.

Table 2
Classification report for SVM

Class	Precision	Recall	F1-score	Support
NO	0.58	0.66	0.62	265
YES	0.61	0.53	0.57	266
Accuracy	–	–	0.62	531
Macro avg	0.60	0.60	0.59	531
Weighted avg	0.60	0.60	0.59	531

4.3. Naive Bayes

Naive Bayes achieved an accuracy of 0.595 and an F1-score of 0.589. The precision for "YES" (0.602) is higher than for "NO" (0.596), but the recall values are balanced. The model's simplicity and assumptions of feature independence may limit its performance, especially if the data violates these assumptions. Despite this, it performs comparably to more complex models like SVM.

Figure 3: Confusion matrix for Naive Bayes

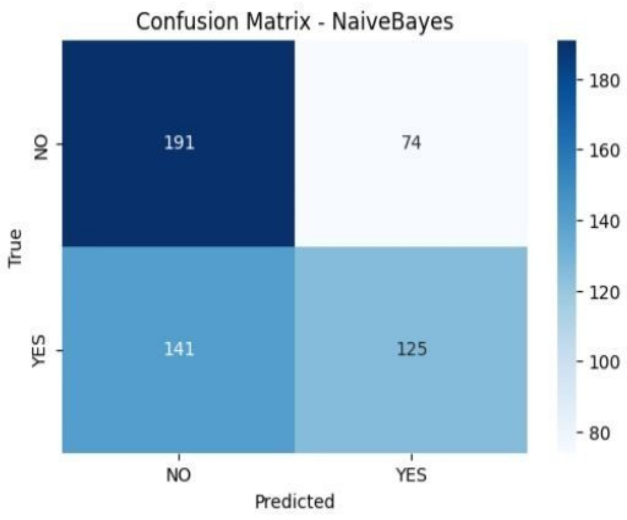


Figure 3 illustrates the confusion matrix for Naive Bayes, while Table 3 presents the corresponding classification report.

Table 3
Classification report for Naive Bayes

Class	Precision	Recall	F1-score	Support
NO	0.58	0.72	0.63	265
YES	0.63	0.47	0.60	266
Accuracy	–	–	0.55	531
Macro avg	0.60	0.60	0.60	531
Weighted avg	0.60	0.60	0.59	531

4.4. Random Forest

Random Forest performed poorly, with an accuracy of 0.571 and an F1-score of 0.571. The low metrics suggest that the ensemble approach did not generalize well for this dataset. This could be due to overfitting or suboptimal hyperparameters. Techniques like feature selection or increasing the number of trees might enhance results.

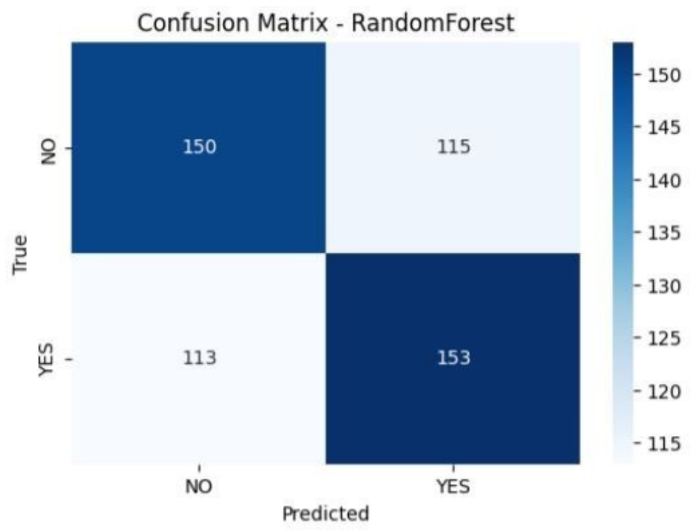


Figure 4: Confusion matrix for Random Forest

Figure 4 illustrates the confusion matrix for Random Forest, while Table 4 presents the corresponding classification report.

Table 4
Classification report for RandomForest

Class	Precision	Recall	F1-score	Support
NO	0.57	0.57	0.57	265
YES	0.57	0.58	0.57	266
Accuracy	–	–	0.57	531
Macro avg	0.57	0.57	0.57	531
Weighted avg	0.57	0.57	0.57	531

4.5. LightGBM

LightGBM showed modest performance, with an accuracy of 0.569 and an F1-score of 0.569. The confusion matrix reveals 149 correct "NO" and 153 correct "YES" predictions, with balanced precision and recall. While the results are not outstanding, LightGBM’s efficiency and scalability make it a viable option for larger datasets.

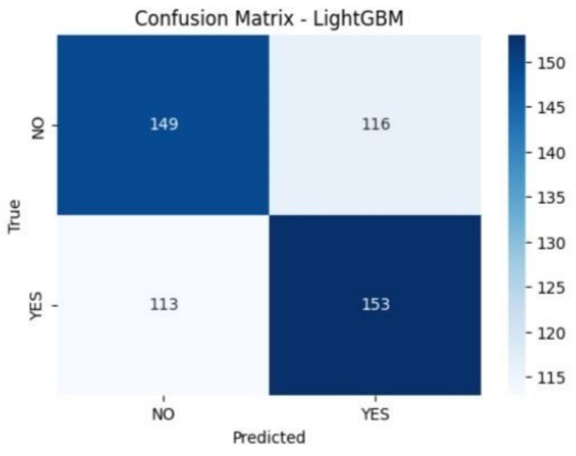


Figure 5: Confusion matrix for LightGBM

Figure 5 illustrates the confusion matrix for LightGBM, while Table 5 presents the corresponding classification report.

Table 5
Classification report for LightGBM

Class	Precision	Recall	F1-score	Support
NO	0.57	0.56	0.57	265
YES	0.57	0.58	0.57	266
Accuracy	–	–	0.57	531
Macro avg	0.57	0.57	0.57	531
Weighted avg	0.57	0.57	0.57	531

4.6. MLP

The MLP model underperformed, with an accuracy of 0.561 and an F1-score of 0.559. The low metrics suggest that the neural network architecture or training process may need optimization, such as adjusting layers, activation functions, or learning rates.

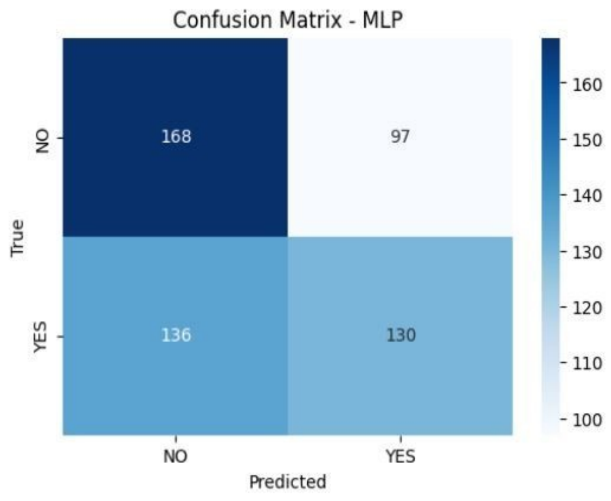


Figure 6: Confusion matrix for MLP

Figure 6 illustrates the confusion matrix for MLP, while Table 6 presents the corresponding classification report.

Table 6
Classification report for MLP

Class	Precision	Recall	F1-score	Support
NO	0.57	0.56	0.57	265
YES	0.57	0.58	0.57	266
Accuracy	–	–	0.57	531
Macro avg	0.57	0.57	0.57	531
Weighted avg	0.57	0.57	0.57	531

4.7. KNN

KNN performed the worst, with an accuracy of 0.55 and an F1-score of 0.60. The confusion matrix shows 128 correct "NO" and 156 correct "YES" predictions, with high misclassifications. The low recall for "NO" (0.72) indicates poor sensitivity to this class. This could be due to the choice of k or distance metric, which may not align well with the data distribution.

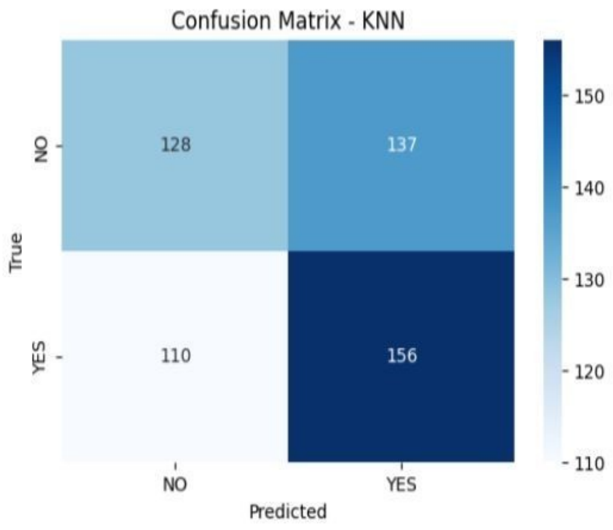


Figure 7: Confusion matrix for KNN

Figure 7 illustrates the confusion matrix for Naive Bayes, while Table 7 presents the corresponding classification report.

Table 7
Classification report for KNN

Class	Precision	Recall	F1-score	Support
NO	0.54	0.48	0.51	265
YES	0.53	0.59	0.56	266
Accuracy	–	–	0.53	531
Macro avg	0.54	0.53	0.53	531
Weighted avg	0.54	0.53	0.53	531

4.8. CatBoost

CatBoost yielded an accuracy of 0.584 and an F1-score of 0.583. The confusion matrix shows 147 correct "NO" and 163 correct "YES" predictions, with moderate misclassifications. The recall for "YES" (0.61) is higher than for "NO" (0.55), indicating a slight bias toward the "YES" class. Hyperparameter tuning or addressing class imbalance might improve its performance.

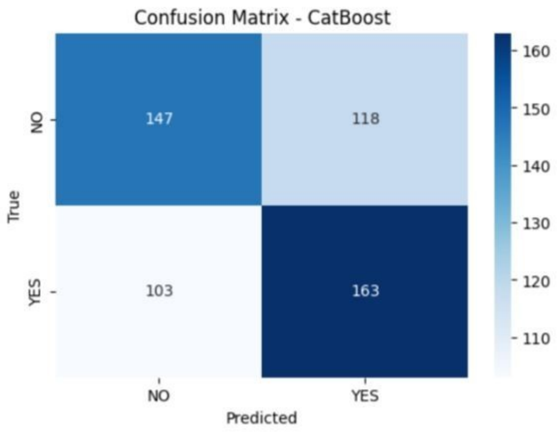


Figure 8: Confusion matrix for CatBoost

Figure 8 illustrates the confusion matrix for Naive Bayes, while Table 8 presents the corresponding classification report.

Table 8
Classification report for CatBoost

Class	Precision	Recall	F1-score	Support
NO	0.59	0.55	0.57	265

YES	0.58	0.61	0.60	266
Accuracy	–	–	0.58	531
Macro avg	0.58	0.58	0.58	531
Weighted avg	0.58	0.58	0.58	531

4.9. XGBoost

XGBoost achieved an accuracy of 0.540 and an F1-score of 0.540, the second-lowest among all models. The results suggest that the default parameters or training setup were ineffective. Parameter tuning or feature engineering might be necessary to leverage XGBoost's potential.

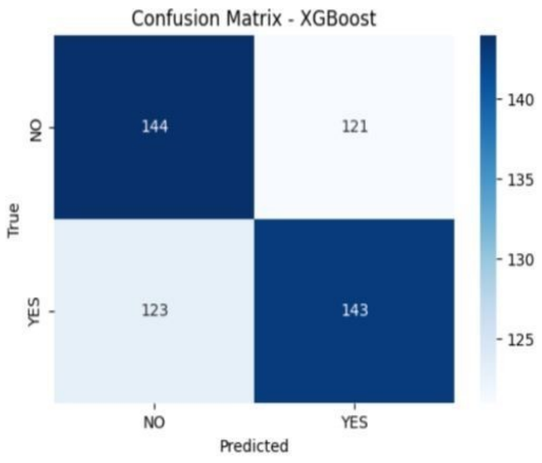


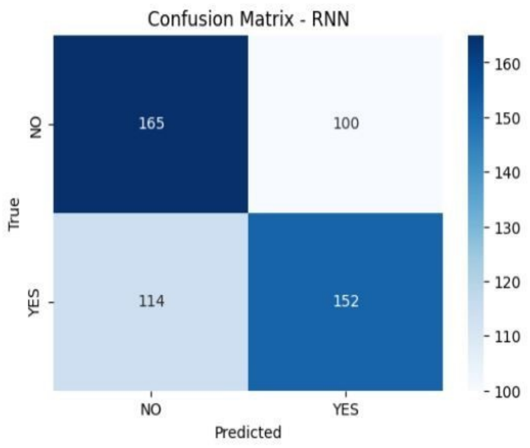
Figure 9: Confusion matrix for XGBoost

Figure 9 illustrates the confusion matrix for XGBoost, while Table 9 presents the corresponding classification report.

Table 9
Classification report for XGBoost

Class	Precision	Recall	F1-score	Support
NO	0.54	0.54	0.54	265
YES	0.54	0.54	0.54	266
Accuracy	–	–	0.54	531
Macro avg	0.54	0.54	0.54	531
Weighted avg	0.54	0.54	0.54	531

4.10. RNN



The RNN model achieved moderate performance, with an accuracy of 0.597 and F1-score of 0.597. The precision and recall values are nearly identical, indicating balanced performance across classes. However, the results are slightly lower than Logistic Regression, which may imply that the sequential nature of the data does not significantly enhance predictions for this task. Further hyperparameter tuning or feature engineering might improve its performance.

Figure 10: Confusion matrix for RNN

Figure 10 illustrates the confusion matrix for RNN, while Table 10 presents the corresponding classification report.

Table 10
Classification report for RNN

Class	Precision	Recall	F1-score	Support
NO	0.59	0.62	0.61	265
YES	0.60	0.57	0.59	266
Accuracy	–	–	0.60	531
Macro avg	0.60	0.60	0.60	531
Weighted avg	0.60	0.60	0.60	531

4.11. FNN

The FNN model showed competitive results, with an accuracy of 0.599 and an F1-score of 0.596. The confusion matrix reveals 137 correct "NO" predictions and 181 correct "YES" predictions, but with higher misclassifications (128 and 85, respectively). The recall for "YES" (0.68) is notably higher than for "NO" (0.52), suggesting the model is more sensitive to the "YES" class. This imbalance could be addressed by adjusting class weights or using techniques like oversampling.

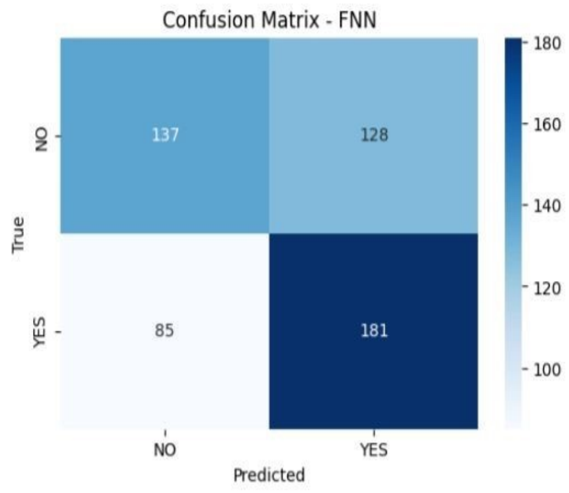


Figure 11: Confusion matrix for FNN

Figure 11 illustrates the confusion matrix for FNN, while Table 11 presents the corresponding classification report.

Table 11
Classification report for FNN

Class	Precision	Recall	F1-score	Support
NO	0.62	0.52	0.65	265
YES	0.59	0.68	0.63	266
Accuracy	–	–	0.60	531
Macro avg	0.60	0.60	0.60	531
Weighted avg	0.60	0.60	0.60	531

4.12. GRU

The GRU model underperformed with an accuracy of 0.589 and an F1-score of 0.589. The low metrics suggest that the GRU's ability to capture temporal dependencies did not translate into better performance for this task. This could indicate that the dataset does not contain significant sequential patterns or that the model requires deeper architecture or more training data.

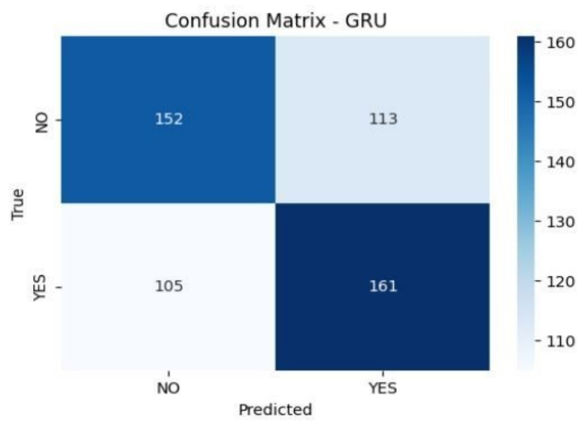


Figure 12: Confusion matrix for GRU

Figure 12 illustrates the confusion matrix for GRU, while Table 12 presents the corresponding classification report.

Table 12
Classification report for GRU

Class	Precision	Recall	F1-score	Support
NO	0.59	0.57	0.58	265
YES	0.59	0.61	0.60	266
Accuracy	–	–	0.59	531
Macro avg	0.59	0.59	0.59	531
Weighted avg	0.59	0.59	0.59	531

4.13. FNN+GRU

The hybrid FNN+GRU model achieved an accuracy of 0.573 and an F1-score of 0.570. The performance is similar to Random Forest, indicating that combining these architectures did not provide a significant advantage. This suggests that the added complexity did not capture additional meaningful patterns in the data.

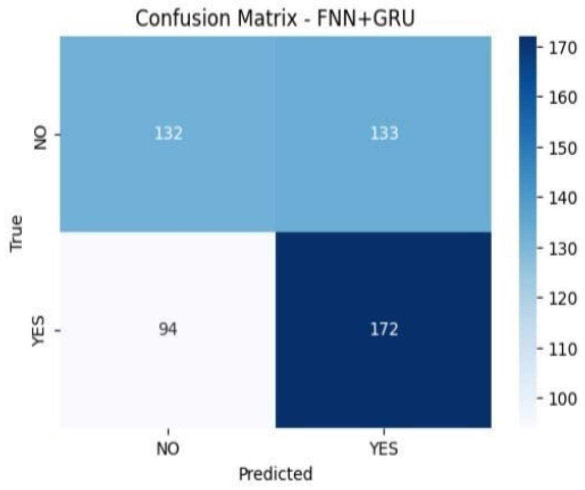


Figure 13: Confusion matrix for FNN + GRU

Figure 13 illustrates the confusion matrix for FNN and GRU, while Table 13 presents the corresponding classification report.

Table 13
Classification report for FNN + GRU

Class	Precision	Recall	F1-score	Support
NO	0.58	0.50	0.54	265
YES	0.58	0.65	0.60	266
Accuracy	–	–	0.57	531
Macro avg	0.57	0.57	0.57	531
Weighted avg	0.57	0.57	0.57	531

4.14. RNN+GRU

The RNN+GRU hybrid model had the lowest accuracy score (0.550) and F1-score (0.550). The confusion matrix shows 148 correct "NO" and 144 correct "YES" predictions, with high misclassifications. This poor performance indicates that the combination of RNN and GRU is not suitable for this dataset, possibly due to overfitting or insufficient training data.

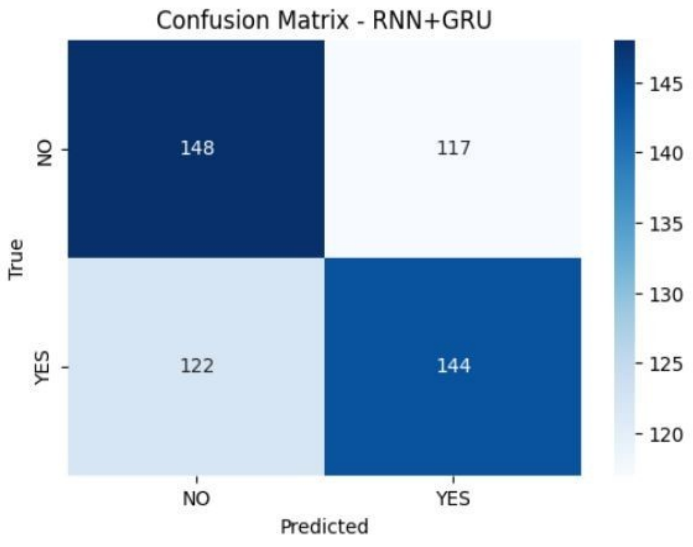


Figure 14: Confusion matrix for RNN + GRU

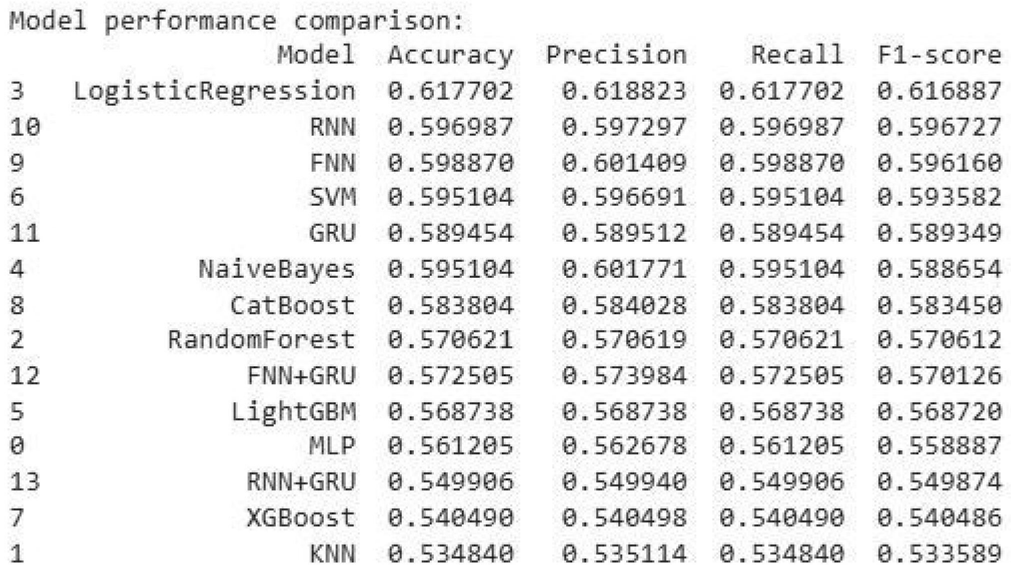
Figure 14 illustrates the confusion matrix for RNN and GRU, while Table 14 presents the corresponding classification report.

Table 14
Classification report for RNN + GRU

Class	Precision	Recall	F1-score	Support
NO	0.55	0.50	0.55	265
YES	0.55	0.54	0.55	266
Accuracy	–	–	0.55	531
Macro avg	0.55	0.55	0.55	531
Weighted avg	0.55	0.55	0.55	531

4.15. Model performance comparison

Logistic Regression emerged as the best performing model, demonstrating robustness and balance across metrics. Simpler models like Naive Bayes and SVM performed comparably to more complex ones, suggesting that the dataset may not benefit significantly from advanced architectures. Hybrid models (e.g., FNN+GRU, RNN+GRU) underperformed, indicating that their added complexity did not translate into better predictions figure 15. Future work could focus on hyperparameter tuning, addressing class imbalance, or exploring alternative feature representations to improve model performance.



```
Model performance comparison:
                  Model  Accuracy  Precision    Recall  F1-score
3    LogisticRegression  0.617702   0.618823  0.617702  0.616887
10                  RNN  0.596987   0.597297  0.596987  0.596727
9                   FNN  0.598870   0.601409  0.598870  0.596160
6                   SVM  0.595104   0.596691  0.595104  0.593582
11                  GRU  0.589454   0.589512  0.589454  0.589349
4            NaiveBayes  0.595104   0.601771  0.595104  0.588654
8              CatBoost  0.583804   0.584028  0.583804  0.583450
2          RandomForest  0.570621   0.570619  0.570621  0.570612
12              FNN+GRU  0.572505   0.573984  0.572505  0.570126
5              LightGBM  0.568738   0.568738  0.568738  0.568720
0                   MLP  0.561205   0.562678  0.561205  0.558887
13              RNN+GRU  0.549906   0.549940  0.549906  0.549874
7               XGBoost  0.540490   0.540498  0.540490  0.540486
1                   KNN  0.534840   0.535114  0.534840  0.533589
```

Figure 15: Overall models performance comparison languages and platforms to assess

5. Discussion

Our experimental findings reveal that traditional models like Logistic Regression and SVM can perform surprisingly well in the task of sexism detection when combined with high-quality contextual embeddings from mT5. Despite their simplicity, these models offer strong generalization and efficiency. In contrast, deep learning and hybrid models did not consistently outperform simpler methods, suggesting that complex architectures are not always advantageous, especially when embeddings already capture rich semantic information. Models like KNN and XGBoost struggled, highlighting challenges related to high-dimensionality and class imbalance. Despite using SMOTE and class weighting, the imbalance between sexist and non-sexist tweets remained a challenge. This points to the need for more advanced data balancing techniques. Finally, ensemble methods showed potential for improving robustness, indicating multiple models may be a promising future direction for enhanced performance.

6. Conclusion and Future Work

This paper presented a comprehensive system for the automatic detection of sexist content in Spanish tweets, developed as part of the EXIST 2025 shared task. Our approach combined the powerful contextual representations from the multilingual mT5 transformer model with a diverse set of traditional machine learning and deep learning classifiers. Experimental results highlighted that Logistic Regression achieved the best overall performance, demonstrating robustness and generalization across classes. Interestingly, simpler models such as Naive Bayes and SVM performed comparably to more complex architectures, indicating that when combined with high-quality embeddings and preprocessing, lightweight models can effectively address the nuanced task of sexism detection. Nevertheless, several challenges remain. The relatively modest performance of deep learning and hybrid models suggests the need for further architectural refinement or the integration of additional linguistic and semantic features. Moreover, the persistent issue of class imbalance underscores the importance of advanced sampling and augmentation techniques to improve model fairness and generalization. For future work, we aim to explore dynamic ensemble strategies that adapt model contributions based on context, incorporate multimodal signals such as images and metadata, and enhance interpretability through explainable AI tools like SHAP and LIME. We also plan to deploy the system in a real-time environment and extend its evaluation to other its cross lingual robustness and applicability.

Declaration on Generative AI

During the preparation of this work, the authors made limited use of ChatGPT, DeepSeek, and Microsoft Copilot. These tools were employed exclusively for grammar and spelling checking, for paraphrasing and rewording sentences in order to improve clarity and style, and for providing occasional code suggestions. All outputs from these tools were carefully reviewed, edited, and validated by the authors to ensure accuracy, originality, and scientific integrity. The authors take full responsibility for the entire content of this publication.

References

[1] Amaan Rizvi, Anupam Jamatia, «NIT-Agartala-NLP-Team at EXIST 2022: Sexism» IberLEF, Spain, 2022. URL: https://anupamjamatia.github.io/assets/pdf/rizvi2022.pdf.

[2] Schutz Mina, Boeck Jaqueline, Liakhovets Daria, Slijepcevic Djordje, Kirchknopf Armin, Hecht Manuel, Bogensperger Johannes, Schlarb Sven, Schindler Alexander, and Zeppelzauer Matthias «Automatic Sexism Detection with Multilingual Transformer Models» IberLEF, Spain., 2021. URL: https://ceur-ws.org/Vol-2943/exist_paper1.pdf.

[3] Sahrish Khan, Gabriele Pergola and Arshad Jhumka, «Multilingual Sexism Identification via Fusion of Large Language Models» Conference and Labs of the Evaluation Forum, France, 2024. URL: https://ceur-ws.org/Vol-3740/paper-99.pdf.

[4] Hannah Kirk, Wenjie Yin, Bertie Vidgen, Paul Röttger, «Explainable Detection of Online Sexism» The 17th International Workshop on Semantic Evaluation (SemEval-2023), Toronto, Canada, 2023. URL: https://aclanthology.org/2023.semeval-1.305/.

[5] Arjumand Younus, Muhammad Atif Qureshi, «A Framework for Sexism Detection on Social Media via ByT5 and TabNet» IberLEF, Coruña, Spain., 2022. URL: https://ceur-ws.org/Vol-3202/exist-paper15.pdf.

[6] AmirMohammad Azadi, Baktash Ansari, Sina Zamani and Sauleh Eetemadi, «Bilingual Sexism Classification: Fine-Tuned XLM-RoBERTa and GPT-3.5 Few-Shot Learning» Conference and Labs of the Evaluation Forum, Grenoble, France, 2024. URL: https://arxiv.org/pdf/2406.07287.

[7] Zi Yun Yang, Ziqing Zhang, Yisong Miao, «The ELCo Dataset: Bridging Emoji and Lexical Composition» The 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), Torino, Italia, 2024. URL: https://aclanthology.org/2024.lrec-main.1381/?

[8] Rajkiran, «XGBoost vs LightGBM vs CatBoost» Medium, 27 06 2025. [Online]. Available: https://medium.com/%40rajkiranrao205/xgboost-vs-lightgbm-vs-catboost-a-practical-comparison-with-coffee-cats-code-5fab396ed39d.

[9] Dan Wang,Yanbo Shen, Dong Ye,Yanchao Yang, Xuanfang Da and Jingyue Mo, «Evaluation of Scikit-Learn Machine Learning Algorithms for Improving CMA-WSP v2.0 Solar Radiation Prediction» Atmosphere, vol. 15, n° %18, 2024. URL: https://doi.org/10.3390/atmos15080994.

[10] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean,Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, «TensorFlow: A system for large-scale machine learning» 2016. URL: https://arxiv.org/pdf/1605.08695.

[11] Uzair Adamjee, «Introduction to NLTK library in Python» 2020 10 19. [Online]. Available: https://python.plainenglish.io/introduction-to-nltk-library-in-python-6fa729b54ad.

[12] Kunjal Chawhan, «Language Detection in Python Using LangDetect: A Quick Guide» 01 11 2024. [Online]. Available: https://www.decodedigitalmarket.com/language-detection-using-langdetect-library-in-python/?

[13] «SMOTE for Imbalanced Classification with Python» 24 04 2025. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-

techniques/?

[14] Ankur Kumar, «Seaborn: Visualize data beyond matplotlib» 02 09 2024. [Online]. Available: https://techifysolutions.com/blog/seaborn-vs matplotlib/?