Project Online Sexism Detection

Md.Fahim Ul Islam

July 11, 2024

1 Abstract

This paper presents an innovative approach to detecting and classifying sexism in text through a hybrid deep learning model that integrates Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) layers, and a self-attention mechanism. The model leverages the CNN layers to effectively extract local and contextual features from the text, which are crucial for identifying linguistic patterns associated with sexism. The LSTM layers are utilized to capture long-range dependencies in the data, allowing for a nuanced understanding of the sequential and temporal dynamics of language that often convey implicit sexist undertones. The addition of a self-attention mechanism enables the model to prioritize and focus on the most informative parts of the text, enhancing its ability to discern subtle and explicit sexist expressions that may vary greatly in linguistic structure and context. Trained on a comprehensive dataset from the SemEval-2023 dataset for Task A and Task B on the Explainable Detection of Online Sexism, which includes a diverse array of text samples annotated for various forms of sexism, our model demonstrates fair performance with 75% and 76% for the Task A and B in accurately classifying sexist content compared to traditional models. This research contributes to the ongoing efforts in computational linguistics to develop tools for social media platforms and other digital forums to automatically identify and mitigate harmful content, thereby promoting safer and more inclusive online environments.

2 Introduction

Sexism in digital communication is a widespread problem that maintains gender differences and develops a culture of discrimination and exclusion [1, 2]. Recent advances in natural language processing (NLP) have created new opportunities for automatically detecting and addressing such hazardous statements in text. However, the intricacy and context-dependency of sexist language provide particular issues that current models frequently fail to handle adequately.

2.1 Background and Motivation

The insidious character of sexism pervades many aspects of online platforms, influencing both individual and societies [3]. Beyond overt examples, research indicates that sexist language appears discreetly, mixed with colloquialisms and cultural nuances that escape traditional detection methods [5, 6]. This covert sexism fosters gender inequality while also perpetuating negative preconceptions, hampering attempts for inclusivity and advancement. As we navigate the digital landscape, we must remain cautious against such insidious kinds of prejudice and work toward a more equal and courteous online environment for all.

2.2 Contribution

This study describes a novel deep learning architecture that combines Convolutional Neural Networks (CNNs) for local feature extraction, Long Short-Term Memory networks (LSTMs) for temporal dependencies, and a self-attention mechanism to highlight contextually significant features. This novel hybrid model is painstakingly designed to address the complexities of sexist language by first improving the detection of intricate verbal patterns linked with sexism using deep contextual analysis. By combining the strengths of CNNs, LSTMs, and self-attention, the model expertly navigates layers of language complexity to detect and emphasize minor sexism indicators. Furthermore, its adaptability to diverse contexts and robustness to various linguistic expressions of sexism are rigorously validated through extensive testing on the SemEval-2023 dataset, confirming its efficacy in addressing the multifaceted challenges posed by sexist language in contemporary discourse. By addressing these deficiencies, the proposed model seeks to considerably advance the field of automated sexism detection, providing a more dependable tool for social media platforms and other digital forums to nurture gender-free spaces. The outcomes of this work are expected to contribute to ongoing concerns in computational linguistics regarding enhancing model sensitivity to social nuances and detecting bias.

3 Data

The dataset for this study was derived from SemEval-2023 Task 10, "Explainable Detection of Online Sexism." Because of its extensive labeling and diverse textual content, this dataset is ideal for constructing and testing models to detect and classify sexist language. It contains over 14,000 labeled sentences gathered from several web platforms, covering a wide range of circumstances and idioms connected to sexism.

3.1 Task A: Binary Sexism Detection

Labels:

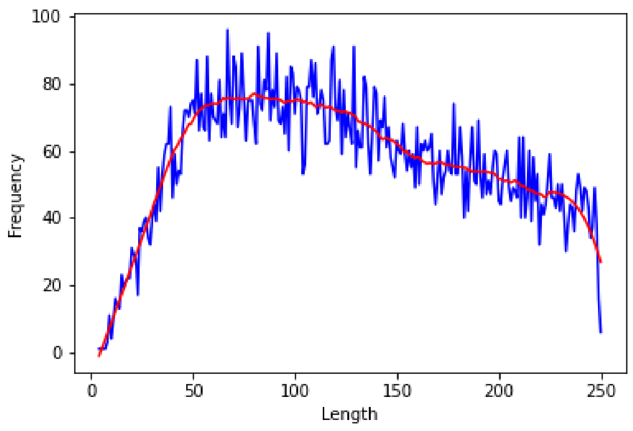

Figure 1: frequency distribution of text lengths

Sexist: This term refers to text that clearly or tacitly supports sexist attitudes or discriminates based on gender. These can range from openly disparaging words to more subtle kinds of bias that might legitimize discrimination. Non-Sexist: Texts that include no sexist content fall into this category. These could include neutral statements, gender conversations that are factual or pro-equality, and unrelated issues. The class distribution is shown in Figure 2.

This task serves as the fundamental binary classification challenge, attempting to determine if a particular text is sexist or not, independent of the type or level of sexism presented.

3.2 Category of Sexism Task B

Labels: 1)Derogation refers to language that demeans someone based on their gender, such as insults or belittlements. 2) Threats and Incitements: Language that not only communicates negative sentiments but also encourages damage or discrimination against people based on their gender. 3)Stereotyping and objectification are statements that promote harmful preconceptions or reduce people to objects or roles based on their gender. 4) Prejudiced Discussions: Texts that justify or support discriminatory practices or views without causing direct damage.

Task B dives further into the various sorts of sexism, requiring the model to not only identify sexist content but also categorize it based on the nature of the sexism displayed. This category helps us grasp the distinct attitudes and actions that must be addressed in different speech contexts.The class distribution is shown in Figure 3.

The dataset's structured approach to sexism classification provides a solid foundation for training and testing NLP models. It aids not only in detecting the presence of sexism, but also in comprehending its forms and contexts,

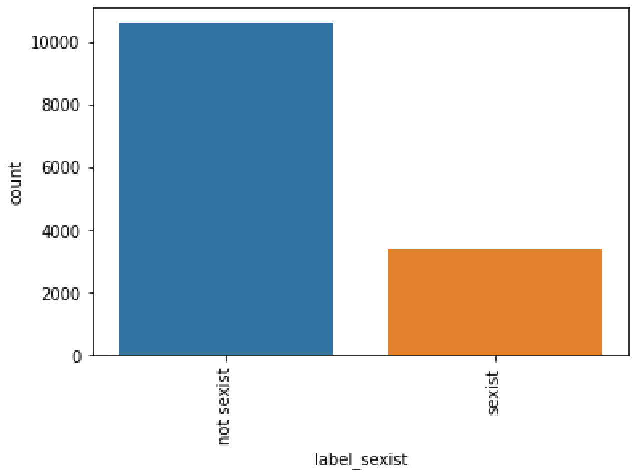

Figure 2: Task A Binary Sexism Detection

which is critical for devising successful solutions in digital interactions. Using this dataset, the proposed model seeks to dramatically improve its capacity to recognize a wide range of sexist expressions, thereby helping to safer and more inclusive online communities.

4 Methodology

4.1 Sentence Transformation

In this study, we offer a unique TensorFlow layer called ”Sentence2Sentence,” which is designed to enable the production of sentence embeddings using the MiniLM-L6 model architecture from the Sentence Transformers library. Our work is inspired by the increased interest in using pre-trained transformer models for a variety of natural language processing tasks. Initially, our strategy requires rigorous initialization of both the tokenizer and the model, with pre-trained weights from the MiniLM-L6-v2 variation used. Following that, we present a systematic methodology for encoding input sentences using the tokenizer and extracting embeddings from these encoded inputs using the MiniLM-L6 model. Our methodology is built around a mean pooling mechanism that is strategically used to generate sentence embeddings. This technique is adept at dealing with the challenges of variable-length inputs, ensuring resilience and semantic coherence while aggregating token-level embeddings. Our methodology is firmly based on current research literature, which highlights the efficacy of transformer-based architectures for encoding textual data into fixed-dimensional representations, with mean pooling emerging as a popular technique for aggregating token embeddings into cohesive sentence representations. Through our

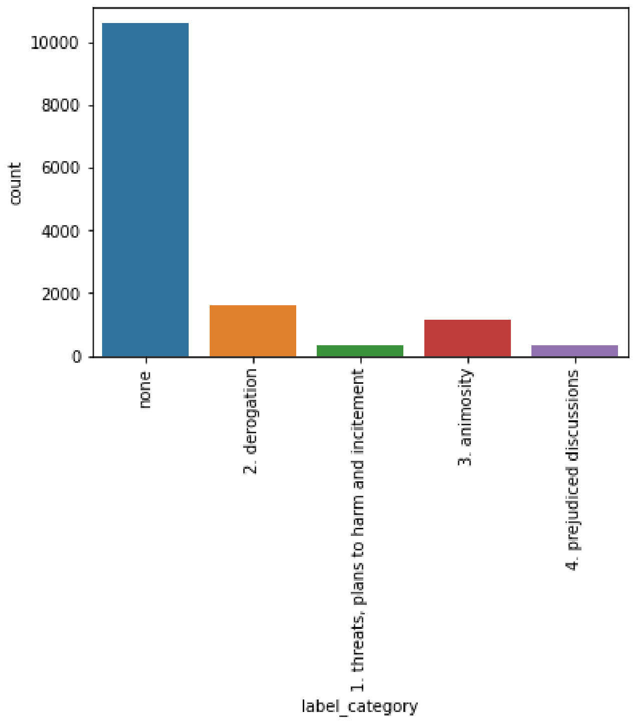

Figure 3: Task B Multiclass Sexism Detection

novel methodology, we contribute to the advancement of sentence representation learning in natural language processing by providing insights and approaches that have broad applicability across many domains and applications. We divided the dataset into three parts: train, test, and validation, with respective ratios of 60%, 30%, and 10%. The dataset was preprocessed to ensure uniformity and optimality for training. This included lowercasing all text, removing non-alphanumeric characters, and replacing URLs and usernames with generic placeholders.

4.2 Model

In this section, we present a comprehensive overview of the network architecture for sentence classification tasks. The architecture is designed to extract hierarchical features from input sentences using a combination of convolutional, recurrent, and attention mechanisms.

The suggested deep learning architecture is precisely designed to handle the complexities of text classification jobs. At its core is an input layer designed to handle textual data, followed by a dedicated Sentence2Sentence layer that aids semantic comprehension. To extract hierarchical characteristics from input sequences, the model employs a sequence of convolutional layers, each carefully set with different filter and kernel sizes. Convolutional layers consisting of 128 filters with a kernel size of 5, 64 filters with a kernel size of 3, and 32 filters with a kernel size of 2 are used in sequence. These convolutional layers are accompanied by max-pooling layers, which are designed to reduce dimensionality and abstract features. The model then incorporates a 64-unit Long Short-Term Memory (LSTM) layer, which allows it to capture the temporal dependencies inherent in sequential data. Notably, a self-attention mechanism is incorporated to improve the model's capacity to identify prominent features, hence increasing discriminative power. Following feature extraction, a flattening layer is used to reduce the multidimensional representation to a one-dimensional vector. This vector is processed through densely connected layers, each with 128 and 64 units, using rectified linear unit (ReLU) activation functions to introduce non-linearity, and dropout layers with a rate of 0.5 to relieve overfitting problems. The last layer, using softmax activation, generates probability distributions across output classes. The model's training includes the binary cross-entropy loss function, which is optimized using the Adam optimizer at a set learning rate. Accuracy and the F1 score are used as evaluation measures, with a focus on the weighted average to overcome class imbalances that occur in classification tasks. The model's comprehensive architecture aims to improve performance in text classification.

1. **Convolutional Neural Network (CNN)**:

- The convolutional layers perform feature extraction from the input sequences. They slide a kernel (filter) over the input data and perform element-wise multiplication followed by summation, which can be represented mathematically as:

$$\text{Convolution}(f(x), w) = \sum_{i} f(x_i) \cdot w_i + b \tag{1}$$

  where $f(x)$ is the input sequence, w is the kernel weights, b is the bias, and i is the index of the elements.

- ReLU activation function is applied after each convolution operation, which introduces non-linearity by mapping negative values to zero:

$$\text{ReLU}(x) = \max(0, x) \tag{2}$$

- MaxPooling layers reduce the dimensionality of the feature maps by selecting the maximum value within each window. This operation can be represented as:

$$\text{MaxPooling}(x) = \max(x) \tag{3}$$

2. **Long Short-Term Memory (LSTM)**:

- LSTM layer is a type of recurrent neural network (RNN) that maintains long-term dependencies in sequential data. It has three gates: input gate (i), forget gate (f), and output gate (o), and a cell state (c_t).

- The computations inside an LSTM cell can be mathematically expressed as:

$$\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\
c_t &= f_t \cdot c_{t-1} + i_t \cdot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \\
h_t &= o_t \cdot \tanh(c_t)
\end{aligned} \tag{4}$$

  Here, x_t is the input at time step t, h_t is the hidden state at time t, c_t is the cell state at time t, W and b are the weights and biases, and σ represents the sigmoid activation function.

3. **Self-Attention Mechanism**:

- Self-Attention mechanism allows the model to focus on different parts of the input sequence, attending to the most relevant information.

- The attention mechanism computes a weighted sum of the input sequence based on the attention scores. The attention scores are computed using a compatibility function, typically a dot product or a learned function.

- Mathematically, the attention mechanism can be represented as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5}$$

  where Q, K, and V are the query, key, and value matrices, respectively, and d_k is the dimensionality of the keys. The softmax function ensures that the attention scores sum up to 1 across the input sequence.

Each of these components contributes differently to the model's ability to understand and process sequential data, ultimately improving its performance on tasks like classification.

5 Experimental Setup

This section details the experimental setup, model training, evaluation methodology, and the results obtained using the novel CNN-LSTM-self-attention model for sexism detection in text.

6 Results

The tables 2 and 3 presented contain performance metrics for two separate activities, identified as Task A and Task B, respectively.

The table displays multiple evaluation metrics, such as loss, accuracy, and F1 score, for both the training set and the validation set, in relation to Task A. The mean values for these indicators are shown, demonstrating the model's

Hyperparameter Type	**Potential Values or Configurations**
Optimizers	Adam: Learning rate (0.001), Beta parameters (beta_1=0.9, beta_2=0.999) RMSprop: Learning rate (0.001), Decay (decay=0.9) SGD: Learning rate (0.01), Momentum (momentum=0.0)
Loss Functions	CategoricalCrossentropy SigmoidFocalCrossEntropy: Gamma (2.0), Alpha (0.25)
Metrics	Accuracy F1Score: Average (micro, macro, weighted)
Callbacks	EarlyStopping: patience=10, monitor='val_loss' ReduceLROnPlateau: factor=0.1, patience=10 ModelCheckpoint: monitor='val_accuracy', save_best_only=True
Learning Rate	Typically starting from 0.001, adjustable in callbacks
Epoch	20
Activation Functions	Dense and LSTM layers often use relu or sigmoid
Layers	LSTM: Units (50-200), Activation (tanh, relu) Conv1D: Filters (32-128), Kernel size (3-7), Activation (relu) Dense: Units (10-1024), Activation (relu, softmax) Embedding: Input dimension, Output dimension Dropout: Rate (0.2-0.5)

Table 1: Typical Hyperparameters and Their Values in Neural Network Training

Table 2: Task A Results

Metric	**Average Value**
loss	0.5632
accuracy	0.7572
f1_score	0.6526
val_loss	0.5532
val_accuracy	0.7574
val_f1_score	0.6528

overall performance in Task A. The model demonstrates notable performance on the training data, with an average loss of 0.5632, an accuracy of 0.7572, and an F1 score of 0.6526. It also shows considerable improvement on the validation set, achieving a loss of 0.5532, an accuracy of 0.7574, and an F1 score of 0.6528.

Similarly, Task B's table presents performance measures such as loss, accuracy, and F1 score for both the training and validation sets. The average values show the model's performance in Task B. In this scenario, the model achieves an average loss of 0.3157, an accuracy of 0.7610, and an F1 score of 0.6595 on the training data. The metrics on the validation set exhibit minor fluctuations, with a loss of 0.2927, an accuracy of 0.7455, and an F1 score of 0.6368.

Overall, these tables assist to completely summarize the performance of the respective models on Task A and Task B, providing insights into their usefulness

Table 3: Task B Results

Metric	Average Value	Additional Column
loss	0.3157	Value1
accuracy	0.7610	Value2
f1_score	0.6595	Value3
val_loss	0.2927	Value4
val_accuracy	0.7455	Value5
val_f1_score	0.6368	Value6

in handling specific classification tasks. The all performance metrics results are shown also in Figure 5 and 6.

7 Conclusion

Our study emphasizes the significance of using appropriate models and regularization techniques for classification problems of various complexity. Future study should address other more advanced strategies, hyperparameter effects on model performance, and data preprocessing techniques. Therefore, explore using a hierarchical strategy to sexism text classification problems to identify performance variances.

References

[1] Ibeji, V. (2023). Gender Discrimination and Female Children's Education in Northern Nigeria: A case study of the Gbagyi of Abuja.

[2] Xie, M., Chao, C. C. (2023). Social Media Portrayal of Housewives and Gender Issues in Chinese Society: A Perspective of Digital Feminism Framework 1. In Mobile Communication in Asian Society and Culture (pp. 24-37). Routledge.

[3] Pérezts, M., Mandalaki, E. (2023). Unsilencing silence on business school sexism: A behind-the-scenes narration on regaining voice. Gender, Work Organization.

[4] Lewandowska-Tomaszczyk, B., Baczkowska, A., Liebeskind, C., Valunaite Oleskeviciene, G., Žitnik, S. (2023). An integrated explicit and implicit offensive language taxonomy. Lodz Papers in Pragmatics, 19(1), 7-48.

[5] Back, H., Piekkari, R. (2024). Language-based discrimination in multilingual organizations: A comparative study of migrant professionals' experiences across physical and virtual spaces. Journal of World Business, 59(3), 101518.

[6] Yu, Y., Si, X., Hu, C., Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. Neural computation, 31(7), 1235-1270.

[7] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

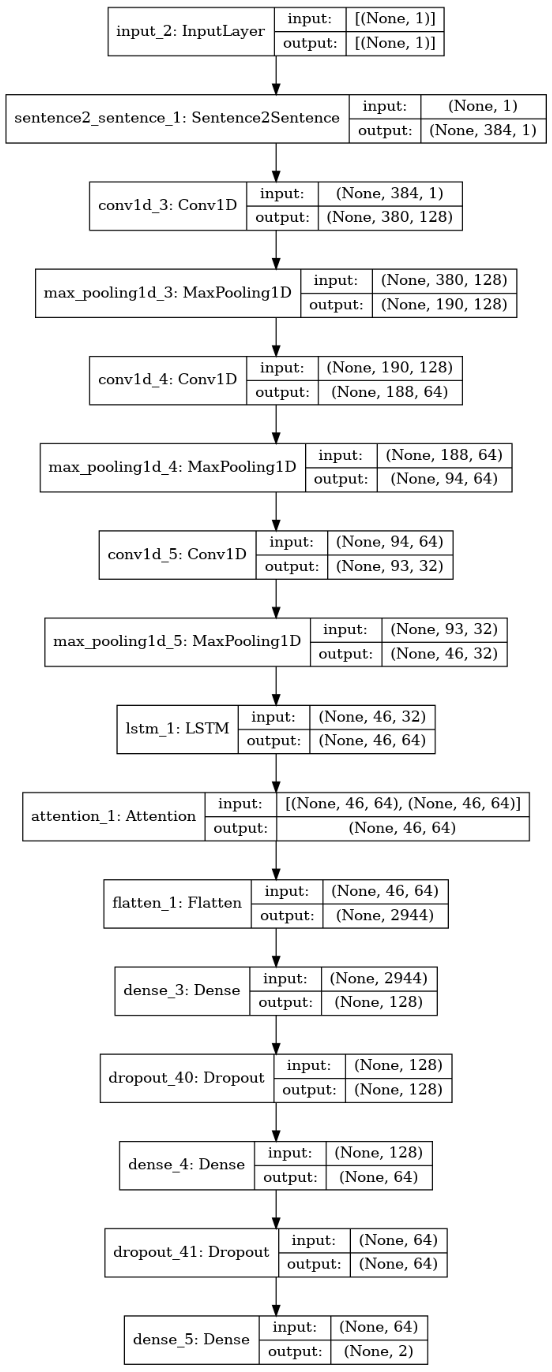

Figure 4: Model Architecture

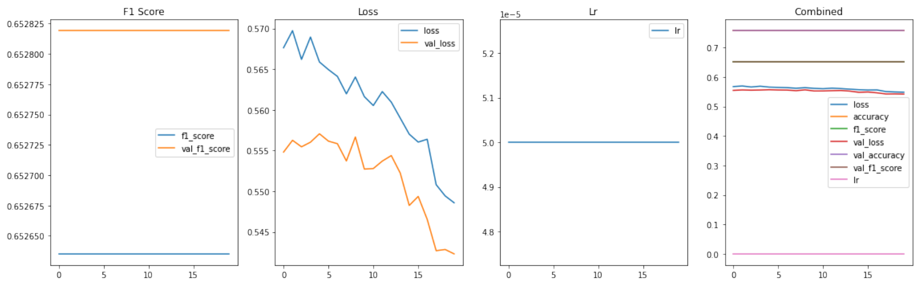

Figure 5: Curve Results for Task A

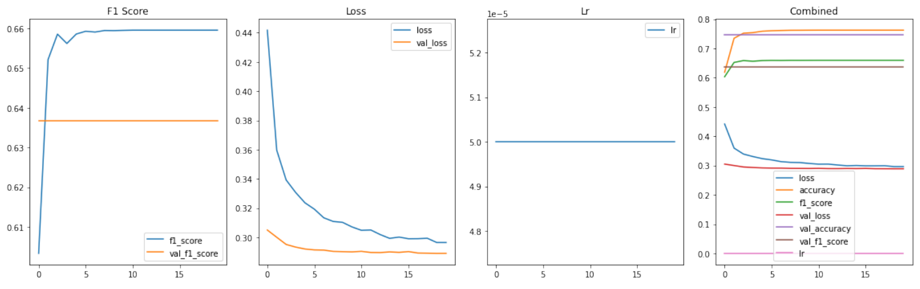

Figure 6: Curve Results for Task B