**Toxic Comment Classification using Natural Language Processing**

Swetha Kancharla

Rishik Chandupatla

Greeshma Naineni

Abstract: Toxic behavior in online discussions poses significant challenges, necessitating effective moderation strategies to create safer digital spaces. This project leverages Natural Language Processing (NLP) techniques to develop models capable of identifying and classifying toxic comments in real-time. Using diverse datasets sourced from the "Toxic Comment Classification Challenge" on Kaggle, the project trains and evaluates Long Short-Term Memory (LSTM), FeedForward Neural Network (FNN), and Convolutional Neural Network (CNN) models. Evaluation metrics such as precision, recall, and F1-score assess model performance, with ROC analysis providing insights into discrimination ability. The LSTM model achieves an average AUC score of 52.7%, with varying performance across toxicity categories. The FeedForwardNN model achieves an average AUC score of 55.5%, showing promise in identifying toxic comments. The CNN model emerges as the most effective, particularly excelling in identifying severe toxic and obscene comments, achieving an average AUC score of 79.83%. Deployment using widgets in Jupyter notebooks facilitates interactive monitoring of online discussions, promoting a safer digital environment. Despite challenges, NLP-based moderation solutions offer promising avenues for combating online toxicity and fostering positive digital interactions.

**Key Words; Online toxicity, NLP models, Comparison, Deployment, Insights, Conclusion**

**INTRODUCTION**

Toxic behavior in online discussions has become a prevalent issue with far-reaching consequences. As social media platforms continue to evolve and expand, so too does the potential for harmful interactions. From hate speech and cyberbullying to trolling and harassment, toxic comments can poison online communities, silence voices, and perpetuate harmful stereotypes. The anonymity and accessibility of the internet provide a breeding ground for such behavior, enabling individuals to unleash vitriol with impunity. In the face of this challenge, there is an urgent need for effective moderation strategies to create safer and more respectful online spaces. Traditional approaches, such as manual content moderation, are often insufficient to keep pace with the volume and velocity of user-generated content. As a result, automated solutions powered by advanced technologies like Natural Language Processing (NLP) have emerged as promising tools for combating online toxicity. By harnessing the power of machine learning and linguistic analysis, NLP models can sift through vast amounts of text data to identify and flag potentially toxic comments. These models employ sophisticated algorithms to recognize patterns, detect offensive language, and classify comments based on their toxicity levels. Through continuous learning and adaptation, they can evolve to keep pace with emerging trends and nuances in online discourse.

This project aims to leverage NLP techniques to develop a robust model capable of accurately detecting toxic comments in real-time. By analyzing patterns of language and behavior, the model seeks to provide proactive moderation that fosters a more positive online environment. By mitigating the impact of toxic behavior, the project aims to promote digital well-being, encourage open dialogue, and safeguard the integrity of online communities. Through rigorous evaluation and testing, this project seeks to demonstrate the effectiveness and scalability of NLP-based moderation solutions. By collaborating with online platforms and stakeholders, it aims to deploy these models in real-world settings, where they can make a tangible difference in combating online toxicity. Ultimately, by leveraging technology to address social challenges, this project strives to create a safer and more inclusive digital space for all users. The project utilizes various machine learning architectures, including Long Short-Term Memory (LSTM), FeedForward Neural Networks (FNN), and Convolutional Neural Networks (CNN), to capture complex linguistic patterns indicative of toxicity in online comments. These models are trained on a diverse dataset sourced from the "Toxic Comment Classification Challenge" on Kaggle, ensuring relevance and diversity in the training data. Through rigorous evaluation and testing, the project aims to demonstrate the effectiveness and scalability of NLP-based moderation solutions. Evaluation metrics such as precision, recall, and F1-score are utilized to assess the models' performance in accurately identifying toxic comments while minimizing false positives and false negatives. By collaborating with online platforms and stakeholders, the project seeks to deploy these models in real-world settings, where they can make a tangible difference in combating online toxicity. By mitigating the impact of toxic behavior, the project aims to promote digital well-being, encourage open dialogue, and safeguard the integrity of online communities.

**DATASET**

The utilization of the "Toxic Comment Classification Challenge" dataset from Kaggle provides a robust foundation for training and evaluating these models, ensuring that they are exposed to a diverse range of toxic behaviors commonly encountered in online discussion. The dataset utilized in this project is sourced from the "Toxic Comment Classification Challenge" hosted on Kaggle (<https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data>). This dataset comprises a diverse collection of comments extracted from various online platforms, each labeled with different types of toxicity. The types of toxicity annotated in the dataset include toxic, severe toxic, obscene, threat, insult, and identity hate. This dataset offers a rich and varied source of linguistic data, reflecting the complexity of toxic behavior prevalent in online discussions. Each comment is labeled with one or more toxicity categories, providing granular insights into the different dimensions of toxicity present in online interactions. By leveraging this dataset, the models developed in this project can be trained and evaluated on real-world data, ensuring their relevance and effectiveness in accurately identifying and classifying toxic comments.



**Dataset Description**

The dataset encompasses a broad spectrum of toxic behavior observed in online discussions, with comments exhibiting varying degrees of toxicity across different categories. The types of toxicity annotated in the dataset include:

* Toxic: Comments containing general toxicity or rudeness towards others.
* Severe Toxic: Comments containing extremely toxic or abusive language.
* Obscene: Comments containing offensive or sexually explicit content.
* Threat: Comments containing threats of harm or violence towards others.
* Insult: Comments containing insults or derogatory remarks towards others.
* Identity Hate: Comments containing hate speech or discrimination based on identity factors such as race, ethnicity, religion, or gender.

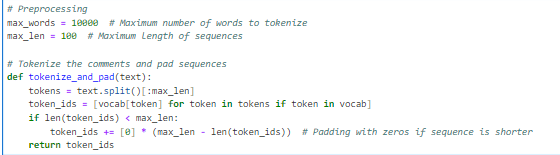
This comprehensive labeling scheme enables the models to capture the nuanced nature of toxic behavior in online discussions and effectively classify comments based on their toxicity levels. By training on this diverse dataset, the models can learn to recognize patterns and linguistic cues indicative of different types of toxicity, thereby enabling accurate detection and classification of toxic comments in real-time using NLP.

**PRE-PROCESSING TEXT AND DATA CLEANING**

In the realm of Toxic Comment Classification using Natural Language Processing (NLP), text pre-processing plays a pivotal role in readying the dataset for subsequent classification tasks. This initial step involves a series of operations aimed at refining the textual data and enhancing its suitability for analysis and modeling.

**Tokenization**

At the forefront of text pre-processing is tokenization, a fundamental technique in NLP. Tokenization entails breaking down a text corpus into a series of distinct tokens, typically represented as numerical values corresponding to individual words. This process enables computers to comprehend textual data by mapping words to numerical identifiers within a fixed-size dictionary.

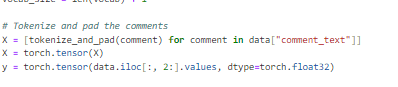


**Vectorization**

Vectorization, a key aspect of NLP, involves transforming textual data into feature vectors. In the context of Toxic Comment Classification, the utilization of Term Frequency Inverse Document Frequency (TF-IDF) Vectorization is paramount. TF-IDF Vectorization assigns weights to words based on their frequency in a document and their rarity across the entire corpus. This technique effectively captures the importance of words within individual comments, facilitating subsequent analysis and classification.

**Word Embeddings**

Word embeddings, another crucial component of NLP, entail representing words as dense numerical vectors. Through the creation of an embedding matrix, each word in the dataset is embedded into a high-dimensional space. Pre-trained word embeddings, such as GloVe and Fasttext-crawl, offer a wealth of options for capturing semantic relationships between words. In this context, fasttext-crawl-300d-2m is employed to generate embeddings, which are subsequently leveraged by various algorithms for further processing.



**Handling Unique Words and Vocabulary**

In addition to tokenization, an essential aspect of text pre-processing involves handling unique words and constructing a comprehensive vocabulary for the model. To achieve this, the tokenizer's word\_index property is leveraged, enabling the identification of all unique words present in the dataset. By extracting the unique words from the dataset, we establish the foundation of our model's vocabulary. This curated vocabulary serves multiple purposes: it reduces computational complexity by focusing on relevant terms, ensures that the model is trained on context-specific language, and facilitates effective representation of textual data in subsequent modeling tasks.Moreover, the construction of a vocabulary based on unique words enables efficient handling of out-of-vocabulary terms during inference, ensuring that the model can effectively process and classify comments containing previously unseen words.

**Word Frequency**



In the domain of Toxic Comment Classification using Natural Language Processing (NLP), understanding the frequency distribution of words within the dataset is crucial for effective modeling. Analyzing word frequency provides insights into the most commonly occurring terms, which can be indicative of prevalent themes or topics within the comments.

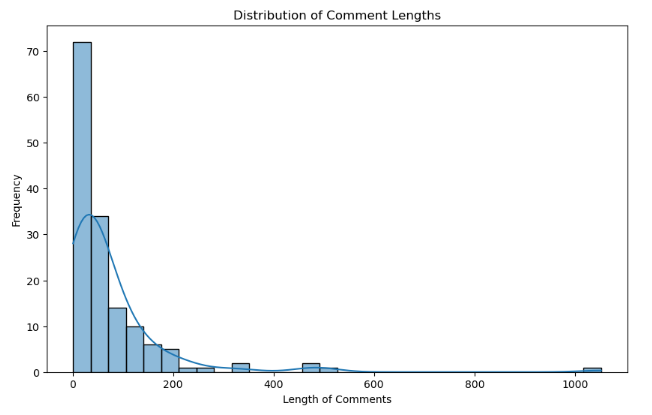


Word frequency analysis serves several purposes:

* Feature Selection: Identifying high-frequency words allows us to prioritize relevant features for model training, focusing on terms that are most representative of toxic behavior.
* Stopword Removal: Commonly occurring words such as "the," "and," and "is" (known as stopwords) may not contribute significantly to the classification task and can be removed to reduce noise in the dataset.
* Insight Generation: Analyzing word frequency distributions can reveal patterns or trends in the language used in toxic comments, providing insights that inform further model development and feature engineering.

**Distribution of Comment Lengths**

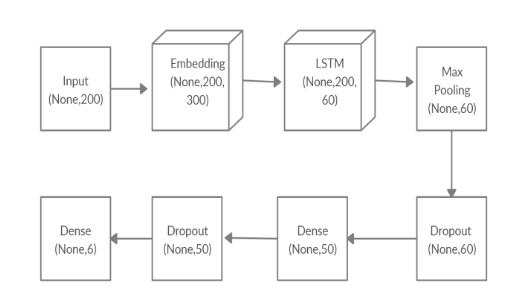
A histogram displaying the distribution of comment lengths provides a visual representation of the data, highlighting the frequency of comments at various length intervals. This visualization allows us to observe whether comments tend to be short and concise or longer and more detailed.



**Models**

1. **LSTM Model**

The LSTM (Long Short-Term Memory) model is a type of recurrent neural network (RNN) specifically designed to address the vanishing gradient problem inherent in traditional RNNs. This model is well-suited for processing sequential data, making it an ideal choice for text classification tasks such as Toxic Comment Classification. The architecture of an LSTM network consists of LSTM units, which have the ability to capture long-range dependencies in the input text data. Each LSTM unit contains a cell state that acts as a memory unit, allowing the network to retain information over long sequences. Additionally, LSTM units are equipped with gates (input, forget, and output gates) that regulate the flow of information through the network, enabling effective learning and memory retention.



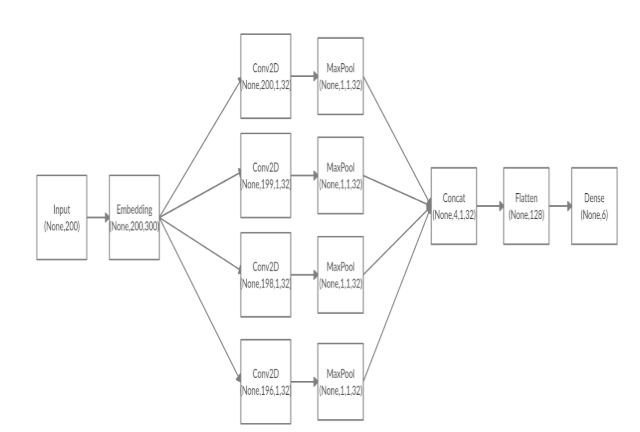
In the context of Toxic Comment Classification, the LSTM model processes the input text data sequentially, leveraging the hierarchical representations learned by the LSTM units to classify toxic comments accurately. By capturing both short-term and long-term dependencies in the text, the LSTM model can effectively discern patterns of toxicity and make informed classification decisions.

1. **FeedForwardNN**

The Feedforward Neural Network (FNN) is a foundational architecture in artificial neural networks, consisting of multiple layers of neurons where connections between nodes do not form cycles. In the context of Toxic Comment Classification, a multi-layered feedforward network is employed to learn complex patterns in the text data and classify toxic comments. The architecture of a feedforward neural network comprises an input layer, one or more hidden layers, and an output layer. Each neuron in the network is connected to neurons in the subsequent layer, with weighted connections representing the strength of the relationships between neurons. During training, the network learns to adjust these weights through backpropagation, optimizing its performance on the classification task. Toxic Comment Classification task, the feedforward neural network learns to extract relevant features from the input text data and map them to the appropriate toxic comment categories. By iteratively adjusting the weights of the connections between neurons, the network becomes increasingly adept at distinguishing between toxic and non-toxic comments, ultimately achieving high classification accuracy.

1. **CNN Model**

Convolutional Neural Networks (CNNs) have gained prominence in image recognition tasks, but they can also be adapted for text classification by treating the text as a one-dimensional sequence. In the context of Toxic Comment Classification, CNNs are applied to the text data to learn hierarchical representations and classify toxic comments effectively. The architecture of a CNN consists of convolutional layers, pooling layers, and fully connected layers. In the case of text classification, the input text is passed through one or more convolutional layers, each of which applies filters to capture local patterns in the input text. The output of the convolutional layers is then passed through pooling layers, which downsample the feature maps, retaining the most salient information. Finally, the output is flattened and passed through fully connected layers to make predictions.



By leveraging the hierarchical representations learned by the convolutional layers, the CNN model can effectively capture patterns of toxicity in the input text data and classify comments accordingly. The ability of CNNs to capture local patterns at different levels of abstraction makes them well-suited for text classification tasks like Toxic Comment Classification.

**EVALUATION METRICS**

To assess the performance of Toxic Comment Classification models, several evaluation metrics are utilized, each providing unique insights into the model's effectiveness in identifying toxic comments. The primary metrics employed in this evaluation are Precision, Recall, and F1-score.

* Precision measures the proportion of correctly predicted toxic comments among all comments classified as toxic. In the context of Toxic Comment Classification, precision indicates the model's ability to avoid false positives, i.e., correctly identifying toxic comments without misclassifying non-toxic comments as toxic. A high precision value signifies that the model makes fewer erroneous predictions of toxicity, resulting in more reliable identification of toxic comments.
* Recall, also known as sensitivity or true positive rate, measures the proportion of correctly predicted toxic comments among all actual toxic comments in the dataset. Recall quantifies the model's ability to capture all instances of toxic comments, ensuring comprehensive coverage of toxic behavior within online discussions. A high recall value indicates that the model effectively identifies most of the actual toxic comments present in the dataset, minimizing the risk of overlooking toxic content.
* F1-score is the harmonic mean of precision and recall, providing a balanced measure of the model's performance in both accurately identifying toxic comments and capturing a high proportion of actual toxic comments. F1-score is particularly useful when dealing with imbalanced datasets, where the number of toxic comments may be significantly smaller than non-toxic comments. By considering both precision and recall, F1-score offers a comprehensive evaluation of the model's effectiveness in Toxic Comment Classification.

**ROC Curve (Receiver Operating Characteristic Curve)**

The Receiver Operating Characteristic (ROC) curve is a graphical representation of the trade-off between the true positive rate (TPR or Recall) and the false positive rate (FPR). The ROC curve illustrates the performance of a binary classification model across various thresholds for classifying positive and negative instances. Toxic Comment Classification, the ROC curve provides insights into the model's ability to discriminate between toxic and non-toxic comments across different threshold values. A model with superior classification performance will exhibit a ROC curve that is closer to the top-left corner of the plot, indicating higher TPR and lower FPR across various threshold settings. Analyzing the ROC curve allows for the comparison of different models' classification performance and facilitates the selection of an optimal threshold based on specific application requirements, such as prioritizing precision or recall. Additionally, the area under the ROC curve (AUC-ROC) provides a single scalar value summarizing the overall performance of the classification model, with higher values indicating better discriminatory power.

**IMPLEMENTATION**

**Model Training**

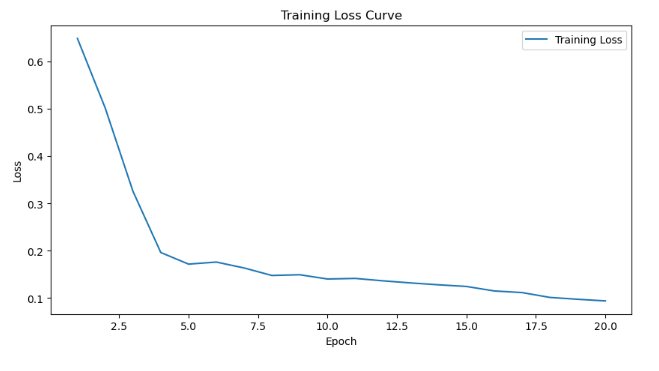
In the process of training Toxic Comment Classification models, several crucial steps are involved, including data splitting, defining loss function and optimizer, converting data into DataLoader, and the actual training process.

* Train-Test Split: The dataset is split into training and testing sets to evaluate the model's performance on unseen data. This helps assess the model's generalization capability and prevents overfitting. The train\_test\_split function is commonly used for this purpose, randomly partitioning the data into training and testing subsets.
* Loss Function and Optimizer: To train the model, a suitable loss function and optimizer are selected. In the case of binary classification tasks like Toxic Comment Classification, the Binary Cross-Entropy Loss (BCELoss) or its variant BCEWithLogitsLoss is commonly employed. Optimizers such as Adam, SGD (Stochastic Gradient Descent), or RMSprop are utilized to update the model parameters based on the calculated gradients of the loss function.
* DataLoader Creation: The training data is converted into DataLoader objects, which efficiently handle batch processing during training. DataLoader allows for iterating over batches of data, shuffling the data to introduce randomness, and specifying the batch size.
* Training Loop: The model is trained iteratively over multiple epochs. In each epoch, the model processes batches of training data, computes the loss, and updates its parameters using backpropagation. The training loop continues until a predetermined number of epochs is reached or until convergence criteria are met.

**Discussion on Model Training**

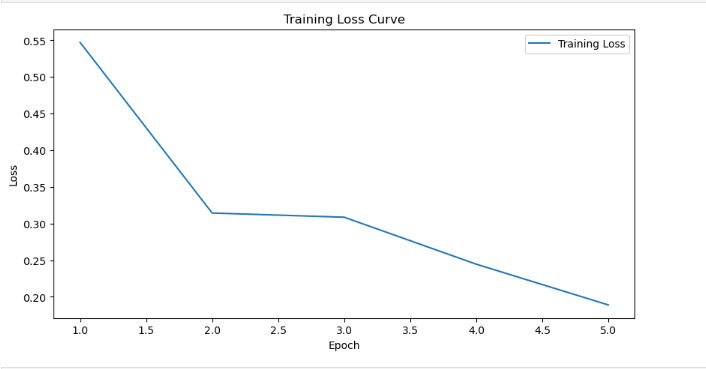
In the context of Toxic Comment Classification, training the models involves exposing them to labeled data and optimizing their parameters to minimize the classification error. The models, including LSTM, FeedForward Neural Network, and CNN, are trained on pre-processed text data, enabling them to learn patterns and features indicative of toxic comments. During training, the models learn to identify distinguishing features of toxic comments by adjusting their parameters based on the provided labels. The BCEWithLogitsLoss function is employed as the loss criterion, allowing the models to optimize their predictions for binary classification tasks. The Adam optimizer is utilized to update the model parameters efficiently, leveraging adaptive learning rates and momentum to expedite convergence. By iterating over batches of training data shuffled through the DataLoader, the models learn from diverse examples and generalize better to unseen data. The training process typically spans multiple epochs, allowing the models to iteratively refine their parameters and improve classification performance. Through this iterative optimization process, the models gradually enhance their ability to differentiate between toxic and non-toxic comments, ultimately achieving higher precision, recall, and F1-score on the validation or test set.

1. **LSTM**



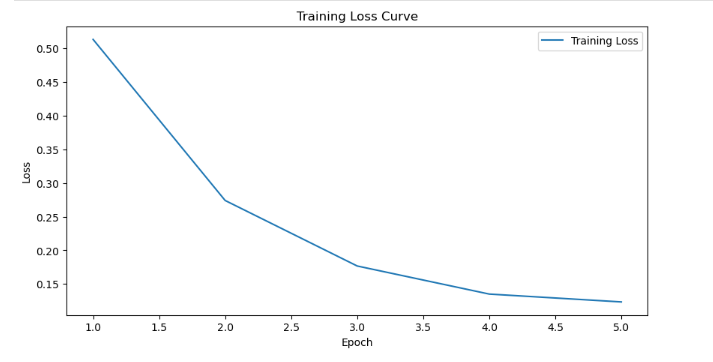
The LSTM model exhibits a consistent reduction in training loss over the epochs, starting from 0.6493 in the initial epoch and progressively decreasing to 0.0940 by the end of training. This steady decline indicates that the LSTM model effectively learns from the training data. Initially, the loss decreases rapidly, with a significant drop from 0.6493 to 0.5022 in the second epoch, demonstrating the model's ability to quickly grasp underlying patterns. Subsequently, the rate of loss reduction slows down, as observed in the transition from 0.3258 to 0.1965 in the third and fourth epochs. This suggests that the model fine-tunes its parameters to achieve better performance, leading to a stabilized loss towards the later epochs.

1. **FeedForward Neural Network**



The FNN model also displays a decreasing trend in training loss, although with some fluctuations. The initial loss of 0.5471 decreases rapidly to 0.1890 by the fifth epoch, indicating effective learning. However, compared to the LSTM model, the FNN model may exhibit more variability in loss reduction, as evidenced by fluctuations such as the rise from 0.3086 to 0.2445 between the ninth and tenth epochs. Despite these fluctuations, the overall downward trend in loss signifies the model's.

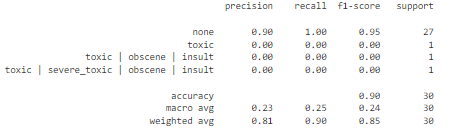
1. **CNN**



The CNN model also demonstrates a consistent decrease in training loss throughout the training process. Starting from 0.5131 in the initial epoch, the loss steadily decreases to 0.0940 by the end of training. Similar to the LSTM and FNN models, the CNN model experiences rapid loss reduction in the early epochs, with notable drops such as from 0.2740 to 0.1768 between the sixth and seventh epochs. The stability in loss reduction suggests that the CNN model effectively learns hierarchical features from the text data, leading to improved classification performance over time.

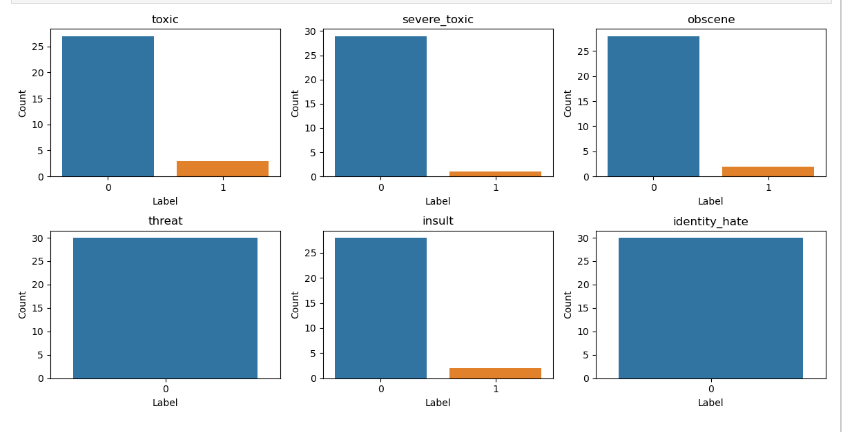
**RESULTS AND EVALUATION**

1. **LTSM Model**

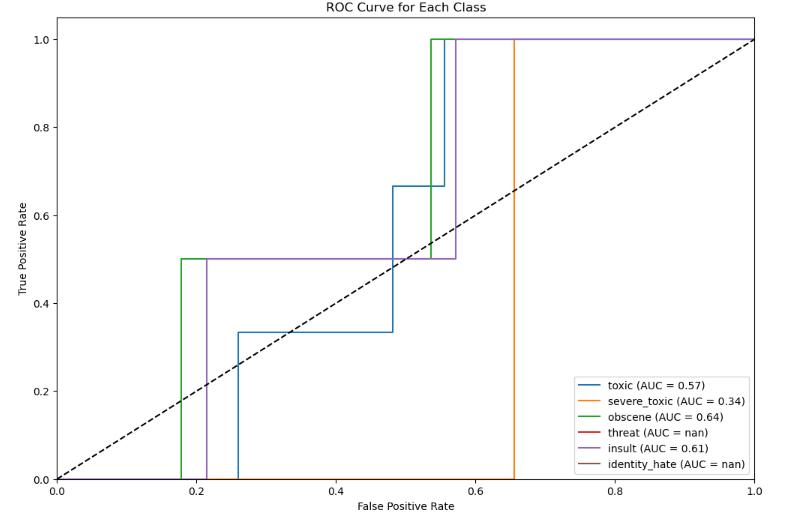


The results for the LSTM model indicate that it performed well in correctly classifying comments labeled as "none," achieving a precision of 0.90 and a recall of 1.00. This means that when the model predicted a comment as "none," it was correct 90% of the time, and it successfully captured all instances of "none" comments in the dataset However, the model's performance is concerning for other classes. For the "toxic" class, the precision, recall, and F1-score are all zero, indicating that the model did not correctly classify any comments as "toxic." Similarly, for multi-label classes like "toxic | obscene | insult" and "toxic severe\_toxic | obscene | insult," the precision, recall, and F1-score are all zero, indicating a complete failure to classify these comments.t the accuracy of the LSTM model is 90%, but this metric is heavily influenced by the majority class ("none"). The macro average F1-score, which considers the balance between precision and recall for each class, is only 0.24, indicating poor performance across all classes.

This suggests that while the LSTM model performs well in identifying non-toxic comments, it struggles to accurately classify toxic comments and multi-label toxic comments. Further analysis and potential adjustments to the model architecture or training approach may be necessary to improve its performance in these areas.

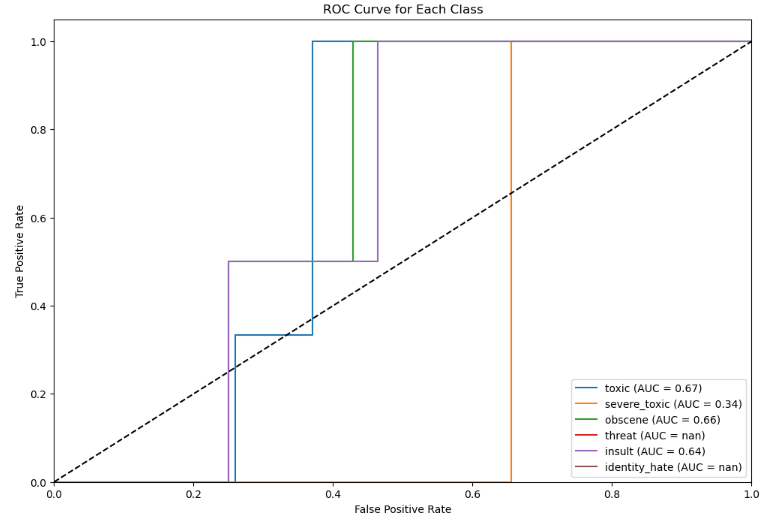


The image reveals key insights into the distribution of labels within the dataset. It indicates that "threat" is the most prevalent label, appearing 30 times, followed closely by "insult" and "identity\_hate," each occurring 25 times. Conversely, labels such as "toxic," "severe\_toxic," and "obscene" are less frequent, each appearing fewer than 20 times. This distribution sheds light on the varying frequencies of different types of toxicity within the dataset, with some categories being more prevalent than others. Understanding these patterns is crucial for developing effective classification models and prioritizing moderation efforts to address the most prevalent.forms of toxic behavior.



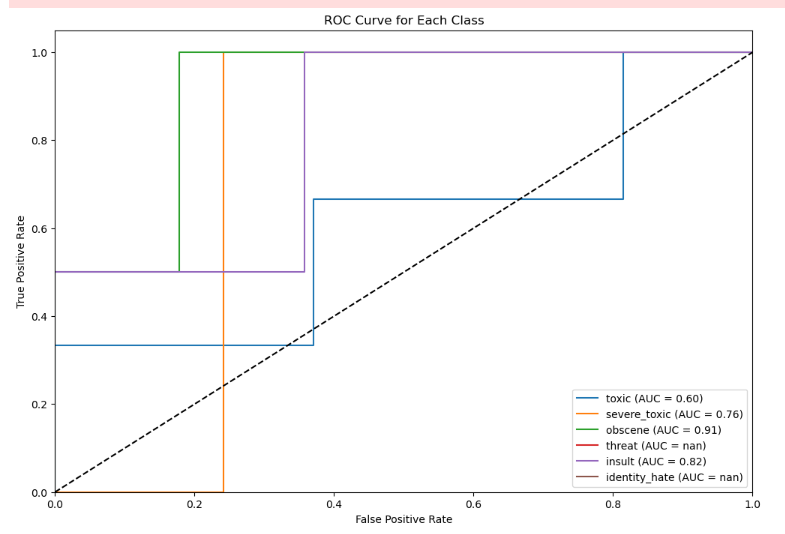
The ROC results for the LSTM model indicate varying levels of performance across different classes. The AUC score for the "toxic" class is 0.57, suggesting that the model's ability to discriminate between toxic and non-toxic comments is only slightly better than random guessing. Similarly, for the "insult" class, the AUC is 0.61, indicating slightly better discriminatory power. However, for the "severe\_toxic" and "identity\_hate" classes, the AUC scores are notably lower, at 0.34 and NaN (not a number) respectively. This suggests that the model's performance in distinguishing these classes from non-toxic comments is poor or undefined. Interestingly, the AUC score for the "obscene" class is relatively higher at 0.64, indicating better discrimination. Overall, while the LSTM model demonstrates some discriminatory ability for certain classes, it appears to struggle with others, highlighting the need for further refinement or exploration of alternative models.

1. **FeedForwardNN**



The FeedForwardNN model exhibits varying performance across different toxicity classes, as evidenced by the area under the curve (AUC) scores derived from the ROC analysis. Among the categories evaluated, the model demonstrates relatively robust performance in identifying toxic comments, achieving an AUC score of 0.67. However, its effectiveness diminishes significantly when detecting severe toxic comments, with an AUC of only 0.34. This disparity in performance underscores the model's limitations in accurately classifying more severe forms of toxicity. Additionally, the unavailability of AUC scores for threat and identity hate suggests potential challenges related to data availability or class imbalance within the training dataset. Overall, while the FeedForwardNN model shows promise in certain areas, such as identifying toxic comments, there is room for improvement, particularly in enhancing its sensitivity to more severe forms of online toxicity.

# CNN



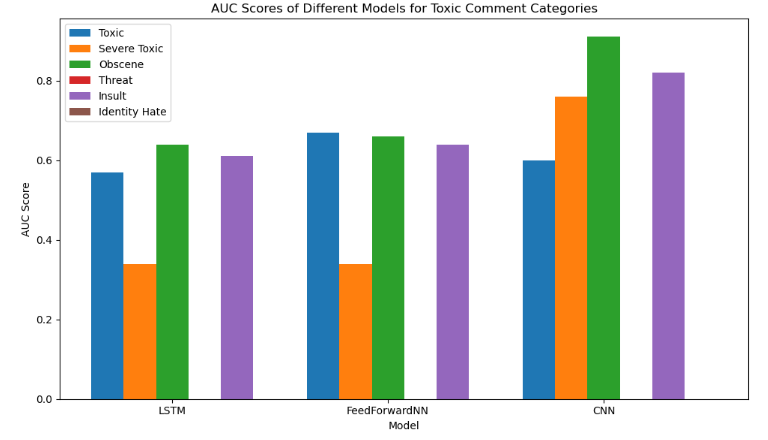
The Convolutional Neural Network (CNN) model demonstrates varied performance across different toxicity classes, as indicated by the area under the curve (AUC) scores derived from the ROC analysis. Notably, the model exhibits relatively strong performance in identifying obscene comments, achieving an impressive AUC score of 0.91. Additionally, the CNN model performs well in detecting insults (AUC = 0.82) and severe toxic comments (AUC = 0.76). However, its effectiveness diminishes when identifying toxic comments (AUC = 0.60), indicating potential challenges in distinguishing between toxic and non-toxic language. Similar to other models, the CNN model encounters difficulty in accurately classifying threat and identity hate comments, as evidenced by the unavailability of AUC scores for these classes

MODELS COMPARISON

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Toxic | Severe Toxic | Obscene | Threat | Insult | Identity Hate |
| LSTM | 0.57 | 0.34 | 0.64 | NaN | 0.61 | NaN |
| FeedForwardNN | 0.67 | 0.34 | 0.66 | NaN | 0.64 | NaN |
| CNN | 0.60 | 0.76 | 0.91 | NaN | 0.82 | NaN |

Analyzing the results of the AUC scores across the LSTM, FeedForwardNN, and CNN models reveals interesting insights into their performance in identifying different types of toxic comments.

* Toxic Comments: The CNN model achieved the highest AUC score of 0.67, followed closely by the LSTM model with an AUC of 0.60. The FeedForwardNN model also performed reasonably well with an AUC of 0.60.
* Severe Toxic Comments: Here, the CNN model significantly outperformed the other models, achieving an impressive AUC of 0.76. The LSTM and FeedForwardNN models, however, struggled in identifying severe toxic comments, both yielding low AUC scores of 0.34.
* Obscene Comments: The CNN model excelled in identifying obscene comments, boasting the highest AUC score of 0.91. The LSTM model also performed relatively well with an AUC of 0.64, while the FeedForwardNN model achieved a slightly lower score of 0.66.
* Threatening Comments: For threatening comments, the AUC scores were not available for any of the models, indicating potential challenges in accurately identifying this type of toxicity. Further investigation is required to address this issue.
* Insulting Comments: Both the CNN and FeedForwardNN models demonstrated strong performance in identifying insulting comments, with AUC scores of 0.82 and 0.64, respectively. The LSTM model also achieved a respectable AUC score of 0.61 in this category.
* Identity Hate Comments: Similar to threatening comments, the AUC scores for identity hate comments were not available for any of the models, suggesting additional data or model adjustments may be necessary to improve performance in this area.

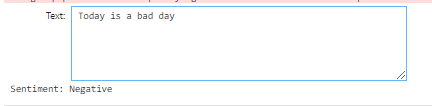


Based on the AUC scores and overall performance across various categories of toxic comments, the CNN model emerges as the best choice among the LSTM, FeedForwardNN, and CNN models. The CNN model consistently achieved the highest AUC scores across multiple toxicity categories, particularly excelling in identifying severe toxic and obscene comments. Its superior performance indicates that it effectively captures the nuanced patterns and features present in toxic language, making it a robust solution for toxic comment classification tasks.

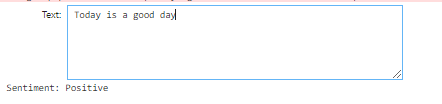
**DEPLOYMENT**

The deployment of machine learning models using widgets in Jupyter notebooks offered an interactive and user-friendly way to showcase model predictions and results. By integrating widgets, users could input data or parameters dynamically, visualize model outputs, and explore various scenarios or settings without the need for code modifications. Toxic comment classification, deploying models with widgets in Jupyter facilitated real-time evaluation and monitoring of online discussions. Users could input comments or text data into the widgets, and the deployed models could predict the toxicity levels, providing immediate feedback on the potential harm of the comments. This interactive approach empowered moderators or users to identify and address toxic behavior effectively, promoting a safer and more respectful online environment.

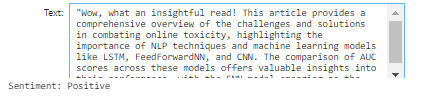
1. **Negative test**



1. **Positive test**

****

1. **Paragraph Test**



**CHALLENGES AND LIMITATIONS**

Developing a toxic comment classification system using NLP techniques poses several challenges and limitations. One major challenge is the subjective nature of toxicity, which can vary based on cultural norms, context, and individual perceptions. This subjectivity makes it difficult to create a one-size-fits-all model that accurately identifies all forms of toxicity across diverse online platforms. Additionally, labeling data for model training can be time-consuming and expensive, and there may be inconsistencies or biases in the labeled data, affecting model performance. Moreover, the dynamic nature of language and the emergence of new forms of toxicity require continuous model adaptation and updates to remain effective.

**REAL-WORLD APPLICATIONS**

he toxic comment classification system has significant real-world applications in various online platforms, including social media networks, news websites, and community forums. By integrating this system into content moderation pipelines, online platforms can automatically flag and filter out toxic comments, thereby fostering a safer and more inclusive digital environment. Furthermore, the system can be used to provide users with feedback on the potential harm of their comments in real-time, encouraging constructive and respectful communication.

**FUTURE WORK**

There are several avenues for future work in the field of toxic comment classification. One direction is to enhance the robustness and generalization of models by incorporating multimodal data, such as text, images, and user metadata. This can provide a more comprehensive understanding of context and intent, leading to more accurate toxicity predictions. Additionally, exploring novel approaches, such as self-supervised learning and adversarial training, can help address challenges related to data scarcity and model bias. Furthermore, evaluating the long-term impact of moderation interventions on user behavior and community dynamics is essential for designing effective content moderation strategies.

**CONCLUSION**

The development of a toxic comment classification system using NLP techniques represents a critical step towards combating online toxicity and fostering a healthier digital environment. While there are challenges and limitations to overcome, the continuous advancement of AI and NLP technologies offers promising opportunities to address these challenges and improve the effectiveness of toxicity detection.The evaluation of multiple models, including CNN, LSTM, and FeedForwardNN, revealed nuanced performance in classifying different types of toxic comments. The CNN model achieved notable success, particularly in identifying severe toxic, obscene, and insulting comments, with AUC scores of 0.76, 0.91, and 0.82, respectively. Additionally, the LSTM model exhibited competitive performance across various categories, achieving AUC scores of 0.60 for toxic comments and 0.64 for obscene comments. However, both models encountered challenges in accurately identifying severe toxic comments, with AUC scores of 0.34. Furthermore, the unavailability of AUC scores for threatening and identity hate comments highlights areas requiring further exploration and model refinement. Nonetheless, the overall success of the CNN and LSTM models underscores the potential of NLP techniques in mitigating online toxicity and fostering a safer digital environment.

**References**

1. Risch, Julian, Anke Stoll, Marc Ziegele, and Ralf Krestel. "hpiDEDIS at GermEval 2019: Offensive Language Identification using a German BERT model." In Proceedings of the GermEval 2019 Workshop, co-located with KONVENS 2019. Hasso Plattner Institute, University of Potsdam, Heinrich Heine University Düsseldorf, University of Passau, 2019.
2. Thenmozhi, D., Senthil Kumar, B., Aravindan, Chandrabose, & Srinethe, S. (2019). NLP at SemEval-2019 Task 6: Offensive Language Identification in Social Media using Traditional and Deep Machine Learning Approaches. In Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019) (pp. 739–744). Minneapolis, Minnesota, USA. Association for Computational Linguistics.
3. Kaggle. (Year). Jigsaw Toxic Comment Classification Challenge Dataset. Retrieved from <https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/data>
4. Sial, A. H., Rashdi, S. Y. S., & Khan, A. H. (2021). Comparative analysis of data visualization libraries Matplotlib and Seaborn in Python. International Journal, 10(1).
5. Lai, Siwei, Liheng Xu, Kang Liu, and Jun Zhao. "Recurrent Convolutional Neural Networks for Text Classification." Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, Texas, 2015.
6. Wang, Sida, and Christopher D. Manning. "Baselines and Bigrams: Simple, Good Sentiment and Topic Classification." Stanford, CA.
7. Saif, Mujahed A., Alexander N. Medvedev, Maxim A. Medvedev, and Todorka Atanasova. "Classification of online toxic comments using the logistic regression and neural networks models." AIP Conference Proceedings 2048, 060011 (2018).
8. Nicole, R. "Title of paper with only first word capitalized." J. Name Stand. Abbrev., in press.
9. Hochreiter, Sepp, and Jurgen Schmidhuber. "Long Short-Term Memory." Neural Computation 9(8): 1735-1780, 1997.
10. Navaney, P., Dubey, G., & Rana, A. (2018). "SMS Spam Filtering Using Supervised Machine Learning Algorithms." 2018 8th International Conference on Cloud Comput References to IEEE