


# A Parallel Neural Network Approach for Offensive Humour Detection

Bhaswanth Reddy I, Vura Abhinav, Karthik M S , Sai Praneeth Reddy A  
Amrita School of Artificial Intelligence, Coimbatore, Amrita Vishwa Vidyapeetham, India  
cb.en.u4aie21121@cb.students.amrita.edu, cb.en.u4aie21177@cb.students.amrita.edu,  
cb.en.u4aie21106@cb.students.amrita.edu, cb.en.u4aie21103@cb.students.amrita.edu

**Abstract**—Offensive humor detection in text is a critical task in natural language processing (NLP) with implications for content moderation and societal well-being. This paper explores a comprehensive approach to offensive humor text classification, leveraging both traditional and cutting-edge techniques. We construct embeddings using frequency-based methods such as n-grams and TF-IDF, alongside prediction-based embeddings like GloVe, FastText, and Word2Vec. Furthermore, we incorporate contextual embeddings from state-of-the-art models like BERT. These embeddings are then fed into both traditional machine learning (ML) models and deep learning architectures, including Neural Tangent Kernel (NTK) networks. Moreover, we propose a novel method called Parallel Neural Network for BERT embeddings, designed to harness the power of BERT while mitigating computational overhead. Through extensive experimentation, we evaluate the performance of each approach on offensive humor classification tasks. Our findings shed light on the efficacy of various embedding techniques in combination with ML and deep learning models for accurately detecting offensive humor in text.

**Index Terms**—Word2Vec, NTK, GloVe, TF-IDF, BERT

## I. INTRODUCTION

Classifying offensive humor presents a significant challenge in the field of natural language processing (NLP) due to the subjective and nuanced nature of humor and offensiveness. This study aims to address this challenge by leveraging a diverse dataset categorized into four levels of offensive humor. By applying traditional machine learning models and parallel neural network architectures, we explore various feature extraction methods, including TF-IDF, GloVe, and Word2Vec embeddings. These techniques enable us to capture the semantic intricacies of the jokes, providing a robust foundation for our classification tasks. Through comprehensive preprocessing, data balancing, and rigorous model evaluation, this study seeks to enhance the accuracy and reliability of offensive humor classification, contributing to advancements in automated content moderation and sentiment analysis.

The literature on offensive humor detection provides various insights and methodologies that inform our approach. The work titled "The Naughtyformer: A Transformer Understands Offensive Humor" [1] introduces a dataset of 92,153 Reddit jokes categorized into four humor subtypes. This study fine-tuned BERT, RoBERTa, DeBERTa, and Longformer models, with DeBERTa-base (Naughtyformer) achieving 87.69% accuracy in humor subtype classification and 92.88

"SemEval-2021 Task 7: HaHackathon, Detecting and Rating Humor and Offense" [2] introduced a dataset of 10,000 texts sourced from Twitter (80%) and the Kaggle Short Jokes dataset (20)

"Detecting Hurtful Humor on Twitter using Fine-Tuned Transformers and 1D Convolutional Neural Networks" [3] presents a dataset of manually labeled tweets. The study extracts embeddings and employs fine-tuning techniques, using 1D Convolutional Neural Networks (CNN) to capture relationships among embeddings. Various models were evaluated, including SVM (F1 score: 0.7500), MLP (F1 score: 0.7327), XGBoost (F1 score: 0.7624), and Random Forest (F1 score: 0.7514). The xlm-roberta-large-spanish model achieved the highest F1 score of 0.8473, demonstrating the effectiveness of combining transformers with CNNs for detecting hurtful humor on Twitter.

"In Unity, There Is Strength: On Weighted Voting Ensembles for Hurtful Humour Detection" [4] introduces a custom dataset to detect humorous content, identify prejudiced groups, and predict the degree of prejudice in Spanish-written tweets. The methodology employs ensembles of state-of-the-art transformer models. Performance evaluations showed F1-scores for BERTc (74.5%), BERTuc (72.1%), RoBERTa (75.9%), DistilBERT (73.6%), BETOc (77.0%), and BETOuc (73.4%). The results highlight the efficacy of using weighted voting ensembles of transformers for hurtful humor detection in Spanish tweets.

"Detection, Classification, and Quantification of Hurtful Humor (HUHU) on Twitter Using Classical Models, Ensemble Models, and Transformers" [5] explores a dataset featuring toxicity, hate speech, emotions, sentiments, irony, and hate speech context in tweets. The study employs classical models, ensemble models, and transformer-based models such as RoBERTa and BETO to detect and quantify harmful humor. Key results include RoBERTa with toxicity features achieving an F1-Macro score of 0.808, RoBERTa with context and most features achieving 0.876, and RoBERTa with a combination of sentiments, emotions, hate, irony, toxicity, and most features achieving 0.821. This work underscores the effectiveness of combining diverse features with advanced models for hurtful humor detection on Twitter.

"Dimensionality Reduction Techniques to Detect Hurtful Humour" investigates a dataset of 2671 tweets, [6] including humorous content and messages with racial, homophobic, and

offensive language targeting various communities. The study primarily uses pre-trained language models like BERT, with top teams incorporating techniques such as task-adaptive and adversarial training. Results for Task 1 (Humor Detection) show an F1-Score of 0.85191, Task 2a (Prejudice Detection) achieved a Macro F1-Score of 0.92411, and Task 2b (Prejudice Degree Inference) yielded an RMSE of 0.75. This research highlights the role of dimensionality reduction techniques combined with advanced models in detecting hurtful humor.

## II. MATERIALS AND METHODS

### A. Dataset Description

The Offensive Humor Dataset, sourced from Kaggle, comprises text samples classified into four levels of offensiveness: mildly offensive, moderately offensive, highly offensive, and extremely offensive. Each entry in the dataset features a joke or humorous statement, annotated based on its perceived level of offensiveness. This dataset, boasting a total of 92,153 jokes, is instrumental for advancing the development and assessment of models dedicated to detecting and categorizing offensive content within humor. Categorized into four distinct types, the dataset encompasses:

Clean Jokes (7,450 examples) Dark Jokes (79,230 examples) Dirty Jokes (5,473 examples) News articles categorized as non-jokes (10,710 examples)

### B. Preprocessing

Preprocessing in Natural Language Processing (NLP) is vital for ensuring data cleanliness and standardization. The dataset faced an initial challenge of imbalance, with classes skewed in distribution, potentially biasing model performance. To mitigate this, both under-sampling and over-sampling techniques were employed, aiding in balancing class representation for more reliable analysis.

Various NLP preprocessing techniques were then applied to refine the dataset. Null value handling removed jokes with missing values, ensuring data integrity. Tokenization segmented text into tokens, while lowercasing standardized text for consistency. Punctuation removal, stop words removal, stemming, and lemmatization enhanced data clarity by eliminating noise and reducing redundancy.

Normalization standardized text, managing variations in spelling and formatting. Additionally, the removal of URLs and emojis focused analysis solely on linguistic content. These preprocessing steps collectively improved dataset quality, facilitating more accurate analyses and enhancing the effectiveness of NLP applications.

### C. Methodology

For Tf-idf Embeddings, we utilized the entire preprocessed dataset as the corpus for calculating TF-IDF values. Each joke was transformed into a vector representation using TF-IDF weighting, wherein higher weights were assigned to terms that were frequent within the document but rare across the entire corpus. These TF-IDF values were initially generated as vectors with a size of 3700. Subsequently, we applied

Truncated SVD to reduce the vector size to 300 during parameter tuning. These optimized TF-IDF vectors were then stored in a CSV file, ready for further analysis and model training.

In the case of n-gram Embeddings with  $n=4$ , we employed the entire preprocessed dataset to generate n-grams. Each joke was transformed into a vector representation using n-gram weighting, capturing sequences of four consecutive terms. Initially, the resulting vectors were of size 69,000. Through parameter tuning, we resized the vectors to a more manageable size of 100 using techniques such as Truncated SVD. This resizing process helped to condense the information while preserving the essential features captured by the n-gram embeddings. Finally, these optimized n-gram vectors were saved in a CSV file for subsequent analysis and model training.

Moving on to Word2Vec Embeddings Extraction, we employed the Skip-Gram model with 300 dimensions and a window size of 5. This model was trained on the preprocessed joke dataset to learn word embeddings, focusing on predicting context words given a target word and capturing semantic relationships between words. For each joke, individual word embeddings were generated to create a 300-dimensional vector representation, encapsulating the overall semantic content. These Word2Vec embeddings were saved in a CSV file for subsequent analysis and model training.

Lastly, we utilized GloVe Embeddings, leveraging pre-trained GloVe embeddings (glove-wiki-gigaword-100) trained on the Wikipedia and Gigaword corpus. These embeddings, capturing semantic relationships between words in a high-dimensional vector space, were loaded into memory. For each joke in the dataset, individual word embeddings are generated a 100-dimensional vector representation, capturing the overall semantic content in the GloVe embedding space. These GloVe embeddings were then saved in a CSV file, facilitating further analysis and model training. For Tf-idf Embeddings, we utilized the entire preprocessed dataset to calculate TF-IDF values. Each joke was transformed into a vector representation using TF-IDF weighting, generating vectors initially sized at 3700. Through parameter tuning with Truncated SVD, the vectors were reduced to 300. These optimized TF-IDF vectors were stored in a CSV file for further analysis and model training.

In the case of n-gram Embeddings with  $n=4$ , we used the entire preprocessed dataset to generate n-grams, initially resulting in vectors sized at 69,000. With parameter tuning and Truncated SVD, vectors were resized to 100, condensing the information while retaining essential features. These optimized n-gram vectors were saved for subsequent analysis and model training.

Moving on to Word2Vec Embeddings, we employed the Skip-Gram model with 300 dimensions and a window size of 5. Individual word embeddings for each joke were generated, resulting in a 300-dimensional vector representation capturing the overall semantic content. These Word2Vec embeddings were saved for further analysis and model training.

Lastly, we leveraged pre-trained FastText embeddings

(fasttext-wiki-news-subwords-300) and GloVe embeddings (glove-wiki-news-subwords-300). These embeddings, capturing semantic relationships, were used to compute joke embeddings, resulting in a 300-dimensional vector representation. These FastText embeddings were saved for subsequent analysis and model training, ensuring robust offensive humor classification. In this study we propose a parallel neural network architecture to detect offense in humor. Our method leverages several pathways in the network to analyze both individual sentences and the entire text. In the first stage, sentences are separated and transformed into numerical representations using BERT sentence embedding. This embedding process is applied to each sentence independently and also to the complete text. Parallel hidden layers within the network then extract mid-level features from each sentence's embedding, capturing aspects like context and sentence type. Additionally, a dedicated pathway analyzes the entire text's embedding, identifying word-level connections that might influence congruity, such as the presence of synonyms or antonyms. The output of the sentence pathway, a vector of size 20, and the whole-text pathway, a vector of size 60, that are obtained after passing through 3 layers of neural network parallelly are concatenated in the fourth layer and continued in a sequential manner to predict the target value. After the parallel substructure of the design, the model integrates the outputs from all pathways through three sequential neural network layers. This combined analysis allows the network to predict the final outcome.

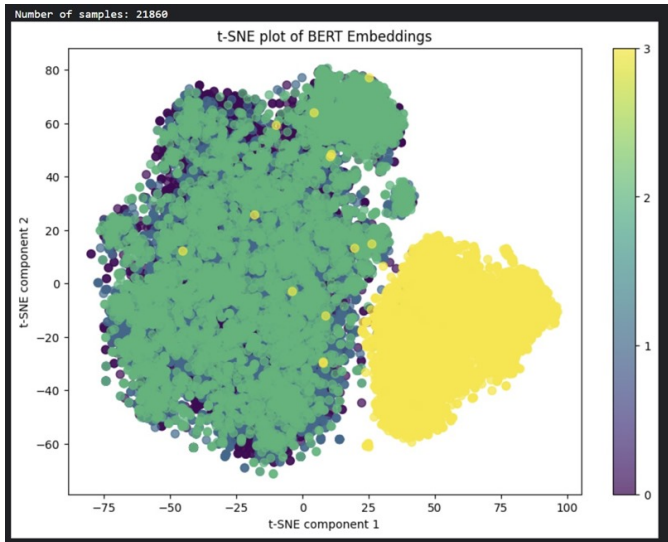


Fig. 1: BERT Embeddings For text

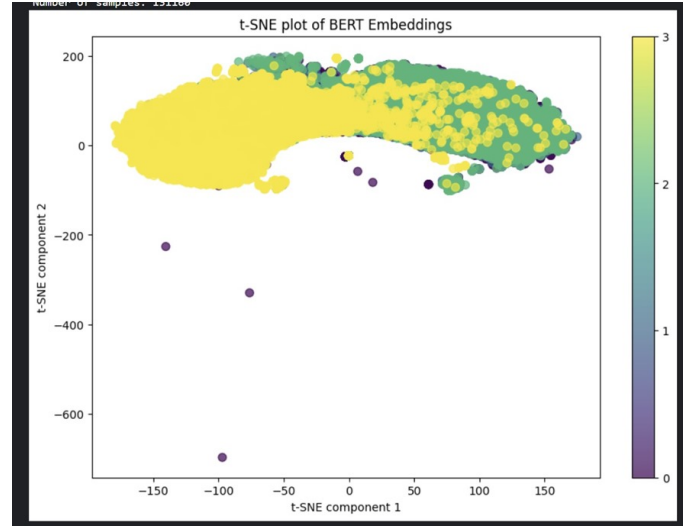


Fig. 2: BERT Embeddings For Sentence

### III. RESULTS

#### n-gram embeddings

Model	Accuracy	Precision	Recall	F1-Score
SVM-rbf	0.56	0.58	0.56	0.56
SVM-poly	0.54	0.56	0.54	0.47
Random Forest	0.59	0.59	0.59	0.59
XG-Boost	0.55	0.56	0.55	0.54
KNN	0.55	0.56	0.55	0.55
NTK	0.55	0.56	0.55	0.55

#### TF-IDF embedding

Model	Accuracy	Precision	Recall	F1-Score
SVM-rbf	0.76	0.76	0.76	0.75
SVM-poly	0.63	0.72	0.63	0.62
Random Forest	0.72	0.72	0.72	0.72
XG-Boost	0.74	0.74	0.74	0.74
KNN	0.62	0.67	0.62	0.62
NTK	0.67	0.71	0.67	0.66

#### Glove Embeddings

Model	Accuracy	Precision	Recall	F1-Score
SVM-rbf	0.72	0.72	0.72	0.72
SVM-poly	0.71	0.72	0.72	0.71
Random Forest	0.71	0.71	0.71	0.71
XG-Boost	0.71	0.71	0.71	0.71
KNN	0.71	0.71	0.71	0.71
NTK	0.71	0.71	0.72	0.71

#### Fasttext Embeddings

Model	Accuracy	Precision	Recall	F1-Score
SVM-rbf	0.77	0.77	0.77	0.77
SVM-poly	0.76	0.76	0.77	0.76
Random Forest	0.74	0.75	0.75	0.74
XG-Boost	0.76	0.76	0.76	0.76
KNN	0.62	0.63	0.62	0.60
NTK	0.73	0.73	0.73	0.73

#### Word2Vec Embeddings

Model	Accuracy	Precision	Recall	F1-Score
SVM-rbf	0.73	0.74	0.74	0.73
SVM-poly	0.73	0.73	0.73	0.73
Random Forest	0.71	0.71	0.71	0.71
XG-Boost	0.73	0.73	0.73	0.73
KNN	0.64	0.64	0.64	0.63
NTK	0.71	0.71	0.71	0.71

#### BERT Embeddings

Class	Accuracy	Precision	Recall	F1-Score
Class-1	0.78	0.73	0.79	0.76
Class-2	0.65	0.69	0.65	0.67
Class-3	0.65	0.67	0.66	0.66
Class-4	1	1	1	1
Overall	0.77	0.77	0.77	0.77

#### IV. CONCLUSION

This study explored offensive humor classification using traditional machine learning models and a parallel neural network approach. Leveraging TF-IDF, n-gram, GloVe, FastText, Word2Vec, and BERT embeddings, models like SVM, Random Forest, KNN, NTK, and XGBoost were trained. With comprehensive preprocessing and data balancing, notable performance gains were achieved. The parallel neural network, adept at capturing intricate nuances, further enhanced accuracy. While traditional models excelled with structured features, the neural network demonstrated superior pattern modeling. This dual approach highlights the complementary roles of traditional and deep learning techniques in advancing NLP tasks.

#### V. FUTURE WORK

In future research, a two-step classification approach is proposed to improve the precision and reliability of humor detection systems by first distinguishing between humor and non-humor texts, and then categorizing the humorous content into specific types: clean jokes, dirty jokes, and dark jokes. The initial phase involves gathering a comprehensive dataset and preprocessing the text to normalize it. Features indicative of humor, such as word frequency, part-of-speech

tags, and semantic embeddings, are extracted. Supervised learning models, including SVM, Random Forest, and deep learning models like CNN or LSTM, are then trained to accurately classify humor from non-humor texts, evaluated using metrics like accuracy, precision, recall, and F1-score. Once humorous texts are identified, the second phase focuses on classifying these texts into distinct joke types. This involves extracting advanced linguistic and semantic features, including sentiment analysis and contextual embeddings, to capture the nuances of different jokes. Multi-class classifiers such as Multinomial Naive Bayes, Multi-Class SVM, or neural network architectures are trained and optimized for this task. The performance of these models is evaluated using macro and weighted F1-scores and validated through k-fold cross-validation. This two-step approach not only aims to enhance the performance of humor detection systems but also contributes to a deeper understanding of humor classification, paving the way for more sophisticated NLP applications.

#### REFERENCES

- [1] L. Tang, A. Cai, S. Li, and J. Wang, "The naughtyformer: A transformer understands offensive humor," *arXiv preprint arXiv:2211.14369*, 2022.
- [2] J. Meaney, S. Wilson, L. Chiruzzo, A. Lopez, and W. Magdy, "Semeval 2021 task 7: Hahackathon, detecting and rating humor and offense," in *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, 2021, pp. 105–119.
- [3] I. Árcos and J. Pérez, "Detecting hurtful humour on twitter using fine-tuned transformers and 1d convolutional neural networks," *IberLEF@ SEPLN*, 2023.
- [4] J. Cruz, L. Elvira, M. Tabernero, and I. Segura-Bedmar, "In unity, there is strength: On weighted voting ensembles for hurtful humour detection," *Iber-LEF@ SEPLN*, 2023.
- [5] H. A. Bonet, A. M. Rincón, and A. M. López, "Detection, classification and quantification of hurtful humor (huhu) on twitter using classical models, ensemble models, and transformers," *Iber-LEF@ SEPLN*, 2023.
- [6] P. S. García and C. M. de la Rosa, "Dimensionality reduction techniques to detect hurtful humour," *Iber-LEF@ SEPLN*, 2023.