Sarcasm Identification Of Dravidian Languages Malayalam and Tamil

Shreyas Karthik, Murari Sreekumar, Kushaal Shyam Potta and Durairaj Thenmozhi

Sri Sivasubramaniya Nadar College Of Engineering, Rajiv Gandhi Salai (OMR), Kalavakkam 603 110, Tamil Nadu, India

Abstract

Sarcasm presents a considerable challenge in the field of sentiment analysis due to its inherently context-dependent nature. On social media, where communications are frequently code-mixed, particularly in Dravidian languages, there is an increasing demand for identifying sarcastic content to ensure effective protection of users from sarcasm disguised as hate or harmful speech. The FIRE 2024 shared task aims to detect sarcasm in Tamil and Malayalam comments collected from the social media platform YouTube. Various traditional machine learning approaches are employed to identify whether the comments contain sarcastic content in Tamil and Malayalam languages. Among these, our logistic regression model achieved a MF Score of 0.68 for Tamil and 0.67 for Malayalam, highlighting its strong fit and capability in addressing the complex nuances of sarcasm detection in code-mixed Dravidian languages.The overall rank we obtained for Tamil and Malayalam set is 7 and 8 respectively.

Keywords

Sarcasm Identification, Traditional Machine Learning Algorithms, Natural Language Processing, Sentiment Analysis, Text Analytics

1. Introduction

Sarcasm is the use of irony to mock or convey contempt, often by saying the opposite of what one actually means. Sarcasm can confuse or alienate those who don't recognize it, leading to misunderstandings or feelings of frustration. It often serves as a rhetorical device to mask hateful sentiments. It allows individuals to express negative opinions while maintaining a façade of humor or irony, making it difficult for both humans and automated systems to identify the underlying hostility. This phenomenon is particularly evident in social media, where users can employ sarcasm to subtly convey contempt or aggression without facing immediate repercussions.[1]

It is crucial to address these aspects of sarcasm in social networks and hence Natural Language Processing research is crucial in providing insights into identifying the comments and classifying them as Sarcastic and Non-Sarcastic. Computational understanding of natural language has been used in addressing issues such as sentiment analysis[2], human behaviour detection, fake news detection[3], question answering and threat detection across different forms of media.

Our research paper presents various innovative solutions contributing to the field of sarcasm identification in significant ways:

- Optimized Approach: The models used in this research like Support Vector Machines (SVM), Logistic Regression, and Random Forest have their hyperparameters tuned to their finest level so that it effectively identifies sarcastic tweets.
- This project can be used for real time applications in social media platforms like Twitter, Instagram, Facebook, LinkedIn etc in order to maintain a healthy and safe online environment.

The task that we have performed in FIRE 2024 [4] [5][6] is Sarcasm Identification of Dravidian Languages (Malayalam and Tamil). In this task, the systems have to decide whether the particular comment is Sarcastic or Not Sarcastic.

Forum for Information Retrieval Evaluation, December 12-15, 2024, India

shreyas2310140@ssn.edu.in (S. Karthik); murari2310237@ssn.edu.in (M. Sreekumar); kushaalshyam2310513@ssn.edu.in (K. S. Potta); theni_d@ssn.edu.in (D. Thenmozhi)
(0000-0003-0681-6628 (D. Thenmozhi)

In this research paper [7] [8], we have discussed the research works that we have done for the task. The rest of the paper is organised as follows: Section 2 presents a literature survey explaining the key theories and concepts, research methodologies and the trends and patterns common in the field of sarcasm identification. Section 3 describes the different datasets used and the task performed. Section 4 talks about the methodology like preprocessing, lemmatization, vectorization and the various models used for our task. Section 5 talks about our results and performance analysis with other teams participating in the task. Finally, in Section 6 we talk about the conclusions and the future prospects of the research work.

2. Related Work

Various works in the field of Sarcasm Identification were studied and diverse methodologies and approach for sarcasm identification and classification were employed to solve this issue. Significant efforts have been made by researchers around the world to develop annotated datasets and apply deep learning models such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). In addition to these, various transformer based models like ROBERT,Distil-Bert have been used as they have consistently provided excellent accuracy in identifying and classifying texts.

Krishnan et.al.[9] undertook a similar research to analyse Tamil and Malayalam code mixed Text and detect sarcasm.The text came from Youtube videos showcasing movie trailers.The text was split into 2 categroies Tamil mixed and Malayalam mixed.They tried around 6 different models such as MLP classifier,Random Forest classifier and Logistic Regression .For vectorization they used Count Vectorizer with n-gram range(1,3) and TF-IDF vectorization . They used a MLP classifier architecture, which had a solitary hidden layer with 128 neurons.A monitoring system was employed to stop training if no performance improvement was seen after 5 iterations, thereby reducing the danger of overfitting,and a maximum of 10 iterations were specified to enable optimal model training.

Bamman.et.al.[10] In this paper based on the tweets provided in twitter they had to perform sarcasm detection.So in this research they tried to tried to exploit logistic regression/maximum entropy, rather than sarcasm detection, using feature extraction include lexical cues and other corresponding sentiment; extralinguistic information of an utterance on Twitter such as the author’s background, audience and communication environment,etc.They used these features in order to do logistic regression.

Chandra.et.al[11] undertook a similar research to analyse Tamil and Malayalam code mixed Text and detect sarcasm. The dataset provided to them is a valuable resource for our research, encompassing code-mixed comments and posts in Tamil-English and Malayalam-English, sourced from social media. While comments and posts may consist of multiple sentences, the dataset predominantly featured an average sentence length of one. Importantly, each comment and post comes with sentiment polarity annotations, reflecting real-world scenarios and challenges associated with class imbalance.For research methodology they used bert-based multilingual trained models.mBERT is built on the transformer architecture which employs self attenuation mechanisms both in encoder and decoder. These models are pre-trained on vast text corpora, including Wikipedia, and have a well-established track record of delivering exceptional performance when fine tuned for various downstream tasks. The architectural design begins with the model taking a special [CLS] token as input, followed by a sequence of words. This input traverses through the layers, with each layer applying self-attention mechanisms and forwarding the results to the subsequent encoder. The output from the final layer of the pre-trained mBERT model serves as the input to a softmax feedforward neural network, a critical component in classifying statements into two categories: Sarcastic or Non-Sarcastic. This neural network generates a probability distribution for each word within the sequence across predefined tags. During prediction, the tag with the highest associated probability is selected as the predicted tag for each word. In the training phase, they carefully tuned specific hyperparameters to guide the learning process effectively. These included a learning rate of 0.01, a batch size of 16, and a maximum of 10 training epochs. These hyperparameters were meticulously optimized to ensure the model’s proficiency in code-mixed language identification and sarcasm detection.

Table 1
Distribution of tweet samples across training, development and testing for each language

Task	Language	No. of samples
Training	Tamil	29570
Validation	Tamil	6336
Testing	Tamil	6338
Training	Malayalam	13189
Validation	Malayalam	2827
Testing	Malayalam	2827

Table 2
Examples of Sarcastic and Non Sarcastic comments

Text	Classification
”Intha padam vantha piragu yellarum Thala ya kondaduvanga.”	”Sarcastic”
”ajth sar mass vasanam ner konda baarvai mass.”	”Non -Sarcastic”

Thenmozhi.et.al[12] used INDIC-Bert and Distil-Bert model on a code mixed Tamil and Malayalam dataset.f sarcasm in Tamil and Malayalam is meticulously prepared, incorporating annotated data. The dataset undergoes tokenization following the WordPiece scheme specific to IndicBERT, and it is subsequently partitioned into distinct subsets for training, validation, and testing. The fine-tuning phase is executed with predefined hyperparameters and employs a cross-entropy loss function. A comprehensive evaluation is then conducted, utilizing precision, recall, and F1-score metrics to assess the model's proficiency in detecting sarcasm within the contexts of Tamil and Malayalam.For Distil-Bert model. In the context of DistilBERT fine-tuning, a dataset containing examples of sarcasm in Tamil and Malayalam is collected and tokenized according to DistilBERT's subword scheme. Subsequently, the dataset is partitioned into separate sets for training, validation, and testing. Hyperparameters play a crucial role in guiding the fine-tuning process, with ongoing validation assessments to monitor the model's performance. Evaluation metrics, including precision, recall, and F1-score, are employed to gauge DistilBERT's effectiveness in identifying sarcasm within the context of Tamil and Malayalam.

.

However, the detection and analysis of more subtle, implicit forms of sarcasm remain under-explored.

3. Dataset Description

This section provides information about the mixed language data in Tamil and Malayalam including the details about the dataset and how we prepared it.In our research we used traditional machine learning models such as Support Vector Machines ,Logistic Regression and Multinomial Naive Bayes and so on. The Tamil dataset has 29570 posts for training ,6336 posts for validating and 6338 posts for testing.The Malayalam dataset has 13189 posts for training,2827 posts for validation and 2827 posts for testing.The dataset contains all three types of code-mixed sentences Inter-Sentential switch, Intra-Sentential switch, and Tag switching. Most comments were written in native script and Roman script with either Tamil / Malayalam grammar with English lexicon or English grammar with Tamil / Malayalam lexicon. Some comments were written in Tamil / Malayalam script with English expressions in between.The objective of this work is to divide the postings into two categories in the datasets for Tamil and Malayalam: Sarcastic and Non-sarcastic.

4. Methodology

We trained the traditional machine learning models such as Support Vector Machine (SVM) [13, 14], Multionmial Naive Bayes and Logistic Regression on the training dataset, evaluated the models on the dev dataset and submitted our runs by applying the ML models on the test dataset.

4.1. Preprocessing

Our first step was to clean the data given in order to improve the performance of the machine learning models:

1. Converting the text to lowercase: This ensures consistency in text data. By doing this the vocabulary size is reduced and it reduces the computational requirements.
2. Removing punctuation marks:They often point to external resources that are not relevant to the context of the text being analyzed.
3. Removing http links and emoticons:These do not contribute to the semantic meaning of the text.
4. Removing twitter mentions like `@username`
5. Removing stop words in english and as well as Tamil to improve the accuracy of the models .
6. Since the training dataset provided had a lot of other text languages such as Arabic ,Telugu,Hindi,etc we needed to remove these texts in order to improve the accuracy of the model. In order to overcome this Polyglot library was used which is a library from python in order to perform nlp tasks.Using this library the texts were classified into their languages and hence accordingly other language texts were removed .
7. In the test dataset since other language texts were present google translate library was used in order to translate the texts into Tamil-English,Malayalam-English respectively . We also meticulously fine-tuned labels for clarity, transforming ”Sarcastic” and ”Non-sarcastic” into ”1” (sarcasm) and ”0” (non-sarcasm), ensuring that our data aligns perfectly with our binary classification task.

Table 3
Preprocessing Results

Text	
Before Preprocessing	`Nte ponno... kidilam... marana waiting`
After Preprocessing	`nte ponno kidilam marana waiting`

4.2. Count Vectorizer using N-grams

CountVectorizer is a feature extraction technique in natural language processing (NLP) that converts text data into numerical vectors, which can be used by machine learning algorithms. When you use CountVectorizer with n-grams, you’re telling it to consider sequences of words (or characters) of a specific length, called n-grams, instead of just individual words.

Example :

1) Unigrams (n=1): Each word is treated as a feature. For example, the sentence ”I love cats” would be split into ”I”, ”love”, and ”cats”.

2) Bigrams (n=2): Pairs of consecutive words are treated as features. For example, ”I love cats” would be split into ”I love” and ”love cats”.

3) Trigrams (n=3): Sequences of three consecutive words are treated as features. For example, ”I love cats” would be split into ”I love cats”.

4.3. Model Evaluation

We have used three models using hard-hard labels such as Sarcastic and Non-Sarcastic. They are:

1. Support Vector Machines: A supervised machine learning algorithm that we used for classification and regression tasks. It operates by creating a decision boundary that separates n-dimensional spaces into classes so that a new data point can be assigned to its relevant category.
2. Logistic Regression: It is a regression model mainly used for classification problems. Logistic regression models the probability that a given input belongs to a particular class. It uses the logistic function, also known as the sigmoid function, to map any real-valued number into the range [0, 1].
3. Mutlinomial Naive Bayes: Multinomial Naive Bayes is a machine learning algorithm that's often used for text classification tasks, like spam detection or sentiment analysis. It's called ”Naive” because it assumes that the features (like words in a text) are independent of each other, which is a simplification. The ”Multinomial” part refers to how it models the data: it counts how often each word appears in the text and uses these counts to predict the category of the text (like whether an email is spam or not). It's simple, fast, and works well when the features are word frequencies or counts.
   For SVM, we have tuned the hyper-parameters like regularization parameter (C) and the kernel parameters, such as the gamma parameter for the radial basis function (RBF) kernel. For Logistic Regression, we have tuned hyper-parameters like the regularization strength (often denoted as C). Regularization techniques such as L1 (lasso) and L2 (ridge) are also tuned to improve model generalization.

5. Results and Performance Analysis

5.1. Performance Analysis

Scikit-learn, also known as sklearn, is an open-source, machine learning and data modeling library for Python. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python libraries, NumPy and SciPy. The sklearn metrics library also provides the classification report for evaluation of the performance of the model. The performance is measured using the following metrics:

1) MF Score: MF score refers to McFadden's pseudo-R^2. The value ranges from 0 to 1, where higher values indicate better performance of the model.

The comparison of our team The Three Musketeers with the other teams is represented in the form of the table below.

Table 4
Ranks based on the MF score of our model in comparison with others for Tamil

Run	Rank	MF1
Code Crafters	6	0.69
The-Three_Musketeers	7	0.68
TextTitans	8	0.61

Table 5
Ranks based on the MF score of our model in comparison with others for Malayalam

Run	**Rank**	**MF1**
Tech_Army_KEC	7	0.67
The_Three_Musketeers	7	0.67
SSN_Language	8	0.62

6. Experimental Results

The below given results are the Macro Average Scores for each models used. The highest score recorded for Tamil language is 0.68 with Logistic Regression model and highest score recorded for Malayalam language is 0.67 with Logistic Regression model.

Table 6
Classification Report - Logistic Regression Tamil

Class	Precision	Recall	F1-Score	Support
0	0.81	0.94	0.87	6485
1	0.70	0.38	0.49	2342
Accuracy			0.79	8827
Macro avg	0.75	0.66	0.68	8827
Weighted avg	0.78	0.79	0.77	8827

Table 7
Classification Report - Logistic Regression Malayalam

Class	Precision	Recall	F1-Score	Support
0	0.85	0.98	0.91	2672
1	0.74	0.26	0.39	625
Accuracy			0.84	3297
Macro avg	0.80	0.62	0.67	3297
Weighted avg	0.83	0.84	0.81	3297

7. Reflections

Through this paper we learnt about important methods in the filed of natural language processing and the steps involved in it .We learnt through this task that Logistic regression is a good model for text classification.Logistic Regression is a simple linear model, which makes it easy to understand and interpret.In the context of text classification, each word (or feature) contributes to the classification decision in a linear fashion, which helps in understanding the importance of specific words in the text.Text datasets, after being transformed into feature vectors (using methods like TF-IDF), are often sparse, meaning most of the feature values are zero (most words do not appear in most documents). Logistic Regression performs well with sparse matrices, making it well-suited for text classification tasks where sparsity is common.

8. Conclusion

Through the scope of this paper, we have explored traditional models for classifying sarcastic and non-sarcastic comments using the dataset provided by FIRE 2024, focusing on Tamil and Malayalam languages. Our findings indicate that Logistic Regression model achieved the highest MF score of 0.68 for Tamil and 0.67 for Malayalam. This research contributes to the field of natural language processing (NLP) by providing valuable insights into addressing content moderation issues on online platforms. Moreover, the model can be deployed in real-world applications to monitor and mitigate sarcastic comments on social platforms. In subsequent research, expanding the model to handle multi-class classification problems, integrating advanced techniques such as attention mechanisms, and exploring further preprocessing strategies can enhance its effectiveness. We hope these efforts will contribute significantly to the moderation and detection of sarcastic comments across various social media platforms.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] S. Frenda, et al., The role of sarcasm in hate speech. a multilingual perspective, in: Proceedings of the doctoral symposium of the xxxivinternational conference of the spanish society for natural language processing (sepln 2018), Lloret, E.; Saquete, E.; Mart´ınez-Barco, P.; Moreno, I., 2018, pp. 13–17.

[2] L. Khan, A. Amjad, N. Ashraf, H.-T. Chang, A. Gelbukh, Urdu sentiment analysis with deep learning methods, IEEE access 9 (2021) 97803–97812.

[3] Z. Khanam, B. Alwasel, H. Sirafi, M. Rashid, Fake news detection using machine learning approaches, in: IOP conference series: materials science and engineering, volume 1099, IOP Publishing, 2021, p. 012040.

[4] B. R. Chakravarthi, N. Sripriya, B. Bharathi, K. Nandhini, S. C. Navaneethakrishnan, T. Durairaj, R. Ponnusamy, P. K. Kumaresan, K. K. Ponnusamy, C. Rajkumar, Overview of the shared task on sarcasm identification of dravidian languages (malayalam and tamil) in dravidiancodemix, in: Forum of Information Retrieval and Evaluation FIRE-2023, 2023.

[5] B. R. Chakravarthi, Hope speech detection in youtube comments, Social Network Analysis and Mining 12 (2022) 75.

[6] B. R. Chakravarthi, S. N, B. B, N. K, T. Durairaj, R. Ponnusamy, P. K. Kumaresan, K. K. Ponnusamy, C. Rajkumar, Overview of sarcasm identification of dravidian languages in dravidiancodemix@fire-2024, in: Forum of Information Retrieval and Evaluation FIRE - 2024, DAIICT , Gandhinagar, 2024.

[7] B. R. Chakravarthi, A. Hande, R. Ponnusamy, P. K. Kumaresan, R. Priyadharshini, How can we detect homophobia and transphobia? experiments in a multilingual code-mixed setting for social media governance, International Journal of Information Management Data Insights 2 (2022) 100119.

[8] N. Sripriya, T. Durairaj, K. Nandhini, B. Bharathi, K. K. Ponnusamy, C. Rajkumar, P. K. Kumaresan, R. Ponnusamy, C. Subalalitha, B. R. Chakravarthi, Findings of shared task on sarcasm identification in code-mixed dravidian languages, FIRE 2023 16 (2023) 22.

[9] D. Krishnan, K. Dharanikota, B. Bharathi, Cross-linguistic sarcasm detection in tamil and malayalam: A multilingual approach., in: FIRE (Working Notes), 2023, pp. 259–269.

[10] D. Bamman, N. Smith, Contextualized sarcasm detection on twitter, in: proceedings of the international AAAI conference on web and social media, volume 9, 2015, pp. 574–577.

[11] S. Chanda, A. Mishra, S. Pal, Sarcasm detection in tamil and malayalam dravidian code-mixed text., in: FIRE (Working Notes), 2023, pp. 336–343.

[12] D. Thenmozhi, Sarcasm detection in dravidian languages using transformer models (2023).

[13] S. L. Salzberg, C4. 5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993, 1994.

[14] T. Pranckevičius, V. Marcinkevičius, Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification, Baltic Journal of Modern Computing 5 (2017) 221.