SARCASM DETECTION
FINAL PROJECT REPORT

Group ML: Enora Barbier, Helene Chen, Tanguy Villain, Elene Wang

ABSTRACT

Sarcasm detection is an essential aspect of modern sentiment analysis across digital platforms. This project addresses the challenge of detecting sarcasm in user-generated text, particularly on Twitter and Reddit, where linguistic nuances and platform-specific constraints complicate accurate classification. We explore a range of machine learning methods, including logistic regression, random forest, SVM, and BERT transformer, to identify sarcastic and non-sarcastic content. The Kaggle-sourced dataset of Reddit comments was complemented by Twitter data to highlight performance differences across platforms.

Early models, such as Logistic Regression and Random Forest, achieved modest accuracy and F1 scores but demonstrated difficulties in capturing the subtleties of sarcasm. SVM offered improved results with balanced precision and recall, but the highest performance emerged from a DistilBERT-based classifier, which attained an accuracy of approximately 76% on Reddit data and 77% on Twitter data despite being trained on only half of the dataset.

Our results underscore the value of transformer-based approaches for robust sarcasm detection and provide insights into platform-specific usage patterns. Future work will investigate fine-tuning strategies, larger training sets, and more advanced sentiment-based features to further enhance detection accuracy across diverse social media platforms.

1 INTRODUCTION

In today's digital communication landscape, sarcasm has emerged as a significant challenge for natural language processing (NLP) systems. The ability to detect sarcasm accurately is crucial for understanding true user sentiment, particularly on social media platforms where sarcastic expressions are commonplace. Our project focuses on developing and evaluating machine learning models for sarcasm detection across different social media platforms, specifically Reddit and Twitter, to understand how platform-specific characteristics influence both sarcastic expression and detection accuracy.

The challenge of sarcasm detection stems from its inherent complexity - sarcastic statements often express the opposite of their literal meaning, requiring sophisticated contextual understanding that even humans sometimes struggle with. This complexity is further amplified in text-based communication where traditional contextual cues such as tone, gesture, and facial expressions are absent. The problem becomes particularly intriguing when comparing sarcasm across different social media platforms, as platform-specific constraints and cultural norms may influence how users express sarcasm.

Our project specifically investigates whether and how sarcastic expression differs between Reddit and Twitter, platforms with distinct characteristics. Twitter's 280-character limit might encourage more direct and concise sarcastic expressions, while Reddit's unlimited character count could facilitate more nuanced and elaborate sarcastic comments. This comparative analysis not only advances our understanding of sarcasm in digital communication but also has practical implications for improving sentiment analysis tools across different social media platforms.

2 Problem Definition

The core challenge we address is the binary classification of text as either sarcastic or non-sarcastic, with a specific focus on how this classification task varies across different social media platforms. Our problem can be formally defined as follows:

Given a text input T from either Reddit or Twitter, we aim to develop a function f(T) $\rightarrow \{0,1\}$ where:

- 0 represents non-sarcastic text
- 1 represents sarcastic text

The key aspects of our problem include:

Contextual Understanding:

- Interpretation relies heavily on conversational context and thread structure
- Previous interactions and response patterns provide essential background
- Platform-specific context (e.g., subreddit themes vs Twitter hashtags)
- Situational awareness influences meaning interpretation

Linguistic and Stylistic Elements:

- Platform-specific linguistic patterns and user behaviors
- Distinctive linguistic cues such as exaggerated language
- Punctuation patterns (e.g., exclamation marks, emojis)
- Impact of text length constraints (Twitter's character limit vs Reddit's unlimited text)
- Sentiment pattern variations across platforms

Sentiment Complexity:

- Contrast between positive language and negative intent
- Understanding the difference between literal and intended meaning
- Capturing subtle nuances in longer Reddit posts vs concise Twitter messages

Model Evaluation and Challenges:

- Assessing performance variations across platforms
- Handling inherent subjectivity (same phrase may appear sarcastic to different readers)
- Analyzing platform-specific error patterns
- Measuring impact of text length and complexity on model accuracy
- Consideration of interpretation ambiguity in evaluation metrics

This paper aims to not only improve sarcasm detection accuracy but also provide insights into how platform characteristics influence the expression and detection of sarcasm in digital communication.

3 Related Work

A dissertation published in 2022 called "A Machine Learning Approach to Text-Based Sarcasm Detection" by Lara I. Novic [1] tests machine learning models on Kaggle dataset containing headlines from the satirical news website The Onion and serious news website Huffpost. It showed that logistic regression and SVM were able to reasonably predict a sarcastic label. However, the models were better at predicting non-sarcastic comments. Our tests also showed solid f1 scores for both our logistic regression model and SVM model. When training and testing the reddit data using the SVM model, non-sarcastic comments showed a slightly higher f1 score than sarcastic comments (0.72 vs

0.69). However, even with a balanced dataset, more comments were labelled as not-sarcastic than sarcastic.

According to the literature review “Sarcasm detection using machine learning algorithms in Twitter: A systematic review” [2], combining multiple algorithms such as CNN-SVM were found to be more effective for sarcasm detection. This sentiment is echoed in the paper “Identification of Sarcasm in Textual Data: A Comparative Study” that states that “hybrid models that include Bidirectional LongTerm Short Memory (Bi-LSTM) and Convolutional Neural Network (CNN) outperform others conventional machine learning as well as deep learning models”.

Sarcasm detection is quite complex especially for sentiment analysis since the sentiment can be opposed to the word used. It is important to detect sarcasm especially on social media platforms as the use of sarcasm does not stop increasing overtime and more and more people are on social media. Plus, posts can be reposted, shared and spread really fast. Since the understanding of sarcasm can be influenced across social media platforms but also other factors such as cultural differences, the use of sarcasm can be misleading. The research titled ”Sarcasm Over Time and Across Platforms: Does the Way We Express Sarcasm Change?” [3] highlights that the expression of sarcasm varies significantly across different social media platforms. The study found that the effectiveness of sarcasm detection models can vary depending on the platform, suggesting that platform-specific nuances play a crucial role in accurately identifying sarcastic content. The overall precision of detection of sarcastic statement is far better on twitter than on Reddit (89.31% vs 55.33%).

The study emphasizes that sarcasm is expressed differently across social media platforms such as Twitter, Facebook and even Reddit. These differences arise from platform-specific conventions, audience expectations, and linguistic norms. For example, Reddit sarcasm may often rely on long-form content and contextual metadata like thread structure, upvotes, and user interactions, while Twitter sarcasm tends to be compact and hashtag-driven. This insight underscores the need for platform-specific models for sarcasm detection, aligning closely with our project’s focus on detecting sarcasm in Reddit comments

The article also discusses how the expression of sarcasm evolves over time, influenced by societal trends and platform changes. While our project does not explicitly focus on temporal changes, this observation supports the importance of tailoring models to current platform usage patterns. By using a dataset derived from recent Reddit comments, our project aligns with this principle, ensuring relevance to contemporary expressions of sarcasm.

4 Methodology

4.1 Preprocessing

4.1.1 Cleaning the data

The two datasets used, Sarcasm on Reddit and Tweets with Sarcasm and Irony, were both sourced from Kaggle. After loading the datasets into Python, rows containing missing comments or tweets were removed to ensure data quality. A custom function, preprocess text, was createdto clean and prepare the text data for modelling.The preprocessing steps included:

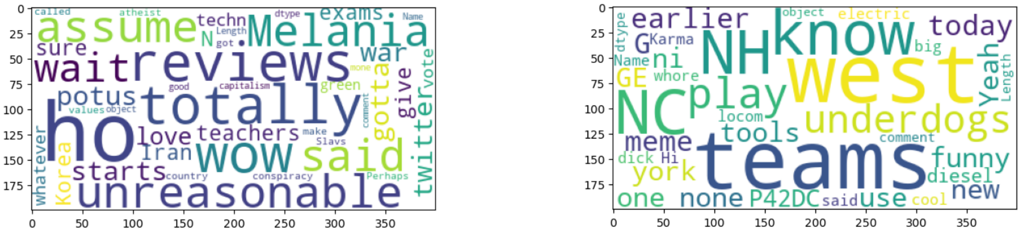
- Converting all text to lowercase for consistency.
- Removing user references and replacing URLs with ¡URL¿.
- Replacing numerical digits with ¡NUM¿ and removing excess whitespace.
- Tokenizing the text using the tokenize function from the nltk package.
- Lemmatizing the words to reduce them to their base forms.
- Finally, the processed tokens were rejoined into complete, cleaned text. Additionally, any missing values were replaced with empty strings to ensure the data remained consistent and suitable for analysis.

4.1.2 Initial Observations of the data

To gain initial insights into the data, word clouds were created for both sarcastic and non-sarcastic comments. These visualizations highlighted the most frequently used words, with the most common terms appearing at the centre of the word clouds.

For sarcastic comments, words like ”assume,” ”wait,” ”totally,” and ”unreasonable” were prominent, likely reflecting the sarcastic tone of the text. Other frequent terms included ”Melania,” ”Twitter,” ”war,” ”country,” ”conspiracy,” and ”Korea,” which are often associated with political or social contexts. Similarly, words such as ”exams” and ”teachers” hinted at educational discussions/contexts, while expressions like ”totally,” ”WOW,” and ”sure” suggested common phrases used ironically.

In contrast, non-sarcastic comments featured terms like ”teams,” ”west,” ”know,” and ”today.” Words such as ”York,” ”NC,” and ”underdogs” were also frequent, indicating that discussions about sports teams, locations, or events were prevalent in this category.



(a) WordCloud data labeled ’sarcastic’

(b) WordCloud data labeled ’non sarcastic’

A comparison between the two categories revealed interesting differences. Sarcastic comments often included exaggerated expressions like ”totally” and ”WOW,” which are typically used ironically, while non-sarcastic comments contained more neutral or context-specific terms.

4.1.3 Sentiment Analysis

A sentiment analysis was performed using the nltk package. A sentiment score for each reddit comment was found. To observe the difference in distribution of sentiment scores for sarcastic comments and non-sarcastic comments, a histogram was plotted. As seen in Figure 1, the distribution in sentiment scores is very similar.

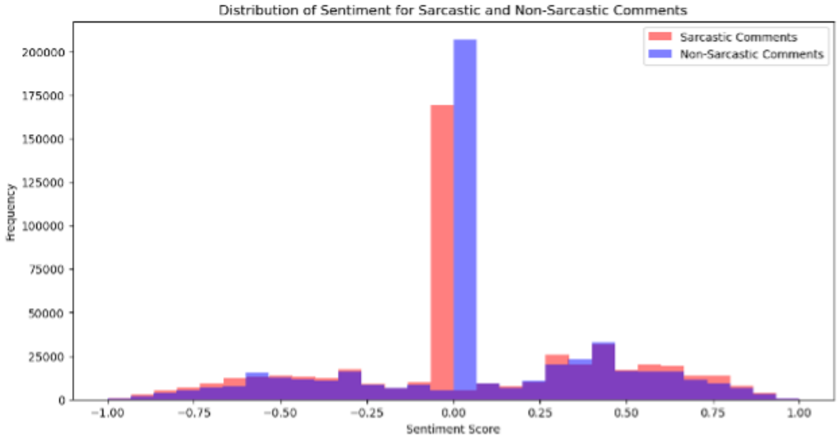


Figure 2: Distribution of Sentiment scores: Sarcastic vs Non-Sarcastic

4.1.4 Vectorization

Machine learning algorithms work with numerical data. Since the comments and tweets are text data, they needed to be vectorized so that they could be then fed into the machine learning model. The

TfidfVectorizer function was used from the sklearn package. The fit transform function was used on the training data in order to allow the vocabulary to be learnt and then for the data to be transformed into vectors. The transform function used the learnt vocabulary to transform the validation data into vectors.

4.2 Models

4.2.1 Logistic Regression: Base model

To classify Reddit comments as sarcastic or non-sarcastic, we employed a Logistic Regression classifier as the foundational model. We choosed it for its simplicity, interpretability, and effectiveness in binary classification tasks. We optimized it using hyperparameter tuning to identify the best-performing parameters. The optimal settings identified were:

- Penalty: L1 regularization ('l1')
- Inverse regularization strength (C): 1.0

These parameters were selected to balance model complexity and prevent overfitting, ensuring that the model generalizes well to unseen data.We then evaluated the performance of the model on 20% of the dataset (202,153 Reddit comments), evenly split between sarcastic and non-sarcastic instances.

4.2.2 Intermediate models

After have trained a logistic regression model on our data, a few models were tested but ended up not performing as well as our base model.

The Random Forest algorithm combines multiple decision trees to reduce bias from class imbalances and overfitting which is why it was explored. The size of the Reddit dataset led to a long training runtime. 200 trees was chosen as it results in a good balance between accuracy and training time. The model was tested with 100 trees but the F1 score decreased from 0.653 to 0.637. In regards to the depth of the tree, a max depth of 6 was chosen to prevent overfitting. A minimal samples split of 3 and minimal samples leaf of 2 was used. For the max features, 'sqrt' was chosen as it is traditionally used for classification.

An ensemble of Logistic Regression, SVM, Random Forest, and XGBoost classifiers combined via a soft voting mechanism was used to build a sarcasm detection model. The dataset was split into training and testing sets, and the ensemble model was trained on the vectorized training data. The model's performance was evaluated using accuracy score and classification report from sklearn.metrics, which provided detailed metrics such as precision, recall, and F1-score. Finally, the trained model was serialized as ensemble model.pkl for future use.

4.2.3 Support machine vector classifier

We also employed a SVM classifier to tackle the sarcasm detection problem, focusing on maximizing the margin between sarcastic and non-sarcastic text in a high-dimensional feature space. SVMs are known for their strong performance on text classification tasks, particularly when combined with effective feature extraction methods. Because sarcasm often relies on subtle linguistic cues scattered throughout a sentence, the SVM's ability to handle sparse, high-dimensional data can be especially advantageous in capturing those nuanced signals.

Our best-performing SVM pipeline used a TfidfVectorizer configured with max_features=10000 and ngram_range=(1,2). By capturing both unigrams and bigrams, we ensured that short phrases, which frequently convey sarcasm, were accounted for. After vectorization, the text was passed to the SVM classifier with a C parameter of 0.1, indicating a preference for a simpler decision boundary to mitigate overfitting. This hyperparameter setting was determined through exhaustive experimentation or grid search, balancing model complexity against generalization. While SVMs typically require careful tuning of both regularization (C) and kernel parameters, we found that even a relatively straightforward linear SVM could achieve competitive results under these settings. Overall, the combination of well-chosen text features and a margin-based learning objective enabled the SVM model to perform robustly on sarcasm detection, highlighting its efficacy as a lightweight yet powerful option alongside more computationally intensive deep learning models.

4.2.4 BERT model

We adopted a DistilBERT-based model for our sarcasm detection task, specifically using the "distilbert-base-uncased" variant from Hugging Face's Transformers library. This choice offers a balanced trade-off between computational efficiency and performance. DistilBERT is a distilled version of the original BERT model that retains a majority of its language understanding capabilities, yet it has fewer parameters. This allows it the be faster to train and less memory-intensive than standard BERT. Given sarcasm's dependence on contextual and semantic subtleties, a transformer architecture that leverages bidirectional self-attention is especially suitable for learning nuanced language patterns without extensive manual feature engineering.

To implement our approach, we employed the AutoTokenizer and AutoModelForSequenceClassification classes, initializing the model with num_labels=2 (sarcastic vs. non-sarcastic). Our training configuration, defined in TrainingArguments, included a warmup phase of 500 steps to stabilize initial gradients and a weight_decay of 0.01 to mitigate overfitting. We used a per_device_train_batch_size of 16 and eval_batch_size of 32, striking a compromise between memory constraints and the need to process enough examples for stable gradient updates. Notably, we enabled fp16 to further reduce memory usage and speed up computations. Our Trainer was set to evaluate and save the model at the end of each epoch, using the F1 metric as the criterion for selecting the best checkpoint.

One practical constraint we faced was limited computational resources, which forced us to train on only half of the available dataset. Although this reduced our overall training time, it also potentially limited the model's exposure to the full spectrum of sarcastic expressions.

4.3 Limitations and Difficulties

One of the most significant challenges we faced was managing the sheer size of the Reddit dataset. While having a large corpus can be advantageous for model training. More data often leads to better generalization but it also introduces substantial computational overhead. Even with efficient data-loading mechanisms, training on a million of different comments demands significant processing power and memory resources. This constraint influenced our model selection, as certain approaches like XGBoost became impractical due to exceedingly long training times. Additionally, large datasets exacerbate hyperparameter tuning complexities, since every trial involves processing vast amounts of data, thereby limiting the number of experiments we could feasibly conduct. Nevertheless, the advantage of having a large dataset is that it helps reduce overfitting, as it provides a broader coverage of linguistic patterns and thus enhances model robustness.

In some instances, to mitigate these issues, we resorted to training on subsets of the data (50% for the BERT model) to reduce the computational cost. This inevitably impacts the representativeness of the training set, potentially overlooking rare sarcastic patterns that might only emerge in larger samples. Moreover, data preprocessing (tokenization, vectorization, etc.) becomes more time-consuming at scale, increasing the risk of encountering memory bottlenecks or system crashes.

5 Evaluation

5.1 Logistic Regression

In this section, we evaluate the performance of the Logistic Regression model we used as basis for our project.

We evaluated the performance of the model on 20% of the dataset (202,153 Reddit comments), evenly split between sarcastic and non-sarcastic instances. The model achieved an accuracy of 68.84%, indicating that approximately 69% of the predictions made by the classifier were correct. The others evaluation metrics are:

- Precision: 67% of the comments predicted as non-sarcastic and 71% of the comments predicted as sarcastic were correctly classified.
- Recall: The model correctly identified 73% of all actual non-sarcastic and 65% of all actual sarcastic comments.
- F1-Score: The harmonic mean of precision and recall is 70% for the non-sarcastic comments and 67% for the sarcastic ones, reflecting a balanced performance.

5.2 Other intermediate models

n addition to our primary model, we explored several alternative classifiers to assess their effectiveness in sarcasm detection.

The extensive size of our dataset made training the XGBoost model computationally infeasible. Consequently, we were unable to proceed with training and evaluating this classifier.

The SGD classifier achieved an accuracy of approximately 67%. While this performance is on par with our Logistic Regression baseline, it did not offer significant improvements to warrant further consideration.

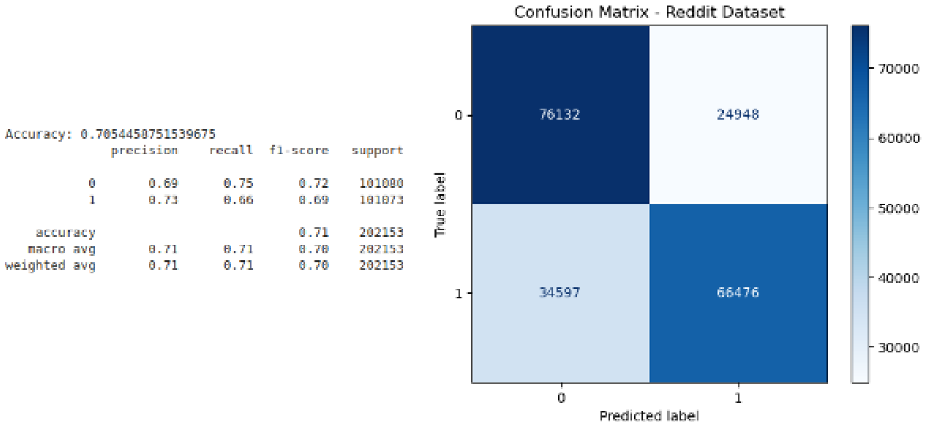
The Random Forest model showed reasonable accuracy, particularly on the Twitter dataset, the lower precision and F1-score indicate challenges in effectively capturing the nuanced patterns of sarcasm. Additionally, the substantial training time required for the large Reddit dataset further limited its practicality for our purposes.

Overall, while models like SGD Classifier and Random Forest provided baseline performances, they fell short compared to the models we are going to explore later. The complexity and subtlety inherent in sarcasm detection necessitate more sophisticated approaches that can capture contextual and linguistic nuances, which ensemble tree-based models struggled to achieve in this project.

5.3 SVM

After exploring additional classifiers to enhance our sarcasm detection performance, the SVM classifier demonstrated the best performance.

We trained the SVM classifier on 80% of our dataset and evaluated it on 20%, evenly split between sarcastic and non-sarcastic instances. The model achieved an accuracy of 70.54%, surpassing the Logistic Regression baseline.



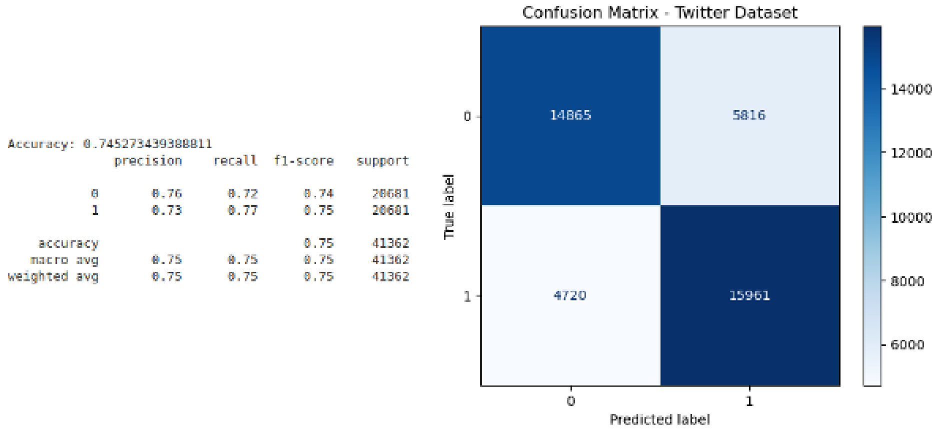
```
Accuracy: 0.7054450751539675
              precision    recall  f1-score   support

           0       0.69      0.75      0.72    101080
           1       0.73      0.66      0.69    101073

    accuracy                           0.71    202153
   macro avg       0.71      0.71      0.70    202153
weighted avg       0.71      0.71      0.70    202153
```

Figure 3: Classification Report Reddit data

To assess the generalizability of the SVM model, we applied the classifier trained on Reddit data to a separate dataset of 41,362 tweets from Twitter, evenly split between sarcastic and non-sarcastic instances. The model achieved an accuracy of 74.53%, indicating robust performance across different platforms.



```
Accuracy: 0.745273439388811
              precision    recall  f1-score   support

           0       0.76      0.72      0.74     20681
           1       0.73      0.77      0.75     20681

    accuracy                           0.75     41362
   macro avg       0.75      0.75      0.75     41362
weighted avg       0.75      0.75      0.75     41362
```

Figure 4: Classification Report Twitter data

The SVM classifier achieved an accuracy of 70.54% on Reddit data, outperforming the other classifier by approximately 2 percentage points. The precision and recall scores for both classes indicate that the SVM model maintains a balanced performance, effectively distinguishing between sarcastic and non-sarcastic comments. Also, when applied to the Twitter dataset, the SVM model achieved a higher accuracy of 74.53%, demonstrating its ability to generalize well across different social media platforms.

The SVM classifier significantly improved our sarcasm detection performance compared to the baseline Logistic Regression and all the other models. Its balanced precision and recall, coupled with strong generalizability across different datasets, make it a robust choice for our classification pipeline. These results highlight the effectiveness of SVM in capturing the nuances of sarcastic language in social media comments. Future work may further enhance performance by exploring feature engineering techniques or integrating ensemble methods.

5.4 BERT CLASSIFIER

Building on the performance of traditional machine learning models, we implemented a BERT-based classifier using the 'distilbert-base-uncased' architecture to further enhance our sarcasm detection capabilities. Due to computational limitations, the BERT model was trained on half of the available dataset. Despite this constraint, the model demonstrated better performance metrics.

This model achieved an accuracy of 76.44% and attained an F1 Score of 76.21% on the reddit data, outperforming SVM and all the models trained before.

To evaluate the model's generalizability, we applied the trained BERT classifier to our separate dataset from Twitter. The model achieved an accuracy of 77.50% and an F1 Score of 77.25% on this dataset, showcasing the model's ability to maintain high performance across different platforms.

The BERT classifier surpassed our first models (around 69% accuracy) and SVM model (70.54% accuracy) by a substantial margin, achieving over 76% accuracy on Reddit data and 77% accuracy on Twitter data.

Leveraging transformer-based architecture, BERT effectively captures the contextual nuances and linguistic patterns inherent in sarcastic language, leading to improved classification performance. Also, the high accuracy and F1 scores on the Twitter dataset indicate that the BERT model generalizes well beyond the Reddit platform, demonstrating its robustness in diverse social media environments.

The BERT classifier represents a significant advancement in our sarcasm detection project, outperforming traditional machine learning models with higher accuracy and F1 scores on both Reddit and Twitter datasets. Its ability to capture complex linguistic features and generalize across different platforms makes it the best-performing model in our current evaluation. Future work will focus on leveraging the full dataset to further enhance the model's performance and exploring fine-tuning strategies to maximize its potential.

6 Conclusion

In conclusion, the BERT model came out as the most effective for sarcasm detection, with the SVM model performing as the second-best alternative. Our results indicate that sarcasm on Twitter is generally easier to detect than on Reddit, a finding consistent with prior literature [3]. One of the primary challenges faced during this project was the long training times required for the large Reddit dataset. With additional time and computational resources, it would be valuable to incorporate more sophisticated sentiment analysis techniques into the model training process. For the BERT model, leveraging the entire dataset and fine-tuning with optimized hyperparameters could yield even stronger performance. As highlighted in the literature [2], neural networks have the potential to create highly effective sarcasm detection models, though such explorations were beyond the scope of this course.

References

[1] Novic, L. I. (2022). A Machine Learning Approach to Text-Based Sarcasm Detection (Master's thesis, City University of New York, CUNY Academic Works). Retrieved from `https://academicworks.cuny.edu/gc_etds/4856`.

[2] Sarsam, S. M., Al-Samarraie, H., Alzahrani, A. I., & Wright, B. (2020). Sarcasm detection using machine learning algorithms in Twitter: A systematic review. *International Journal of Market Research, 62*(5), 578-598. `https://doi.org/10.1177/1470785320921779`.

[3] Bouazizi, M., & Ohtsuki, T. (2022). Sarcasm Over Time and Across Platforms: Does the Way We Express Sarcasm Change? *IEEE Access, 10*, 55958-55987. `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9774408`.