Comparative Analysis of Transformer-Based Models for Sarcasm Detection in News Headlines

Tarik Mohammad Abdel Hadi[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

tarik.hadi@inf.ufrgs.br

Abstract. *This study compared how transformer models perform on sarcasm detection tasks once fine tuned with News Headlines dataset. Three models were fine tuned: RoBERTa, RoBERTa + LSTM + CNN, and GPT-2. The results showed that RoBERTa + LSTM + CNN attained the highest accuracy (0.9047) and highest F1-score (0.8827), outperforming the other two. This suggests that adding sequential and hierarchical feature extraction techniques (LSTM and CNN) enhances the model's ability to capture some complex sarcasm patterns. Future research with more context to the inputs to fine tune the models is highly suggested.*

Keywords. NLP, Text Classification, Sarcasm Detection, RoBERTa, GPT

1. Introduction

Sarcasm is defined by the literal meaning of a sentence differing from the intended meaning. It happens when someone uses words that mean the opposite of what he or she really wants to say. It is usually used in dislike of something, to irritate someone, or to simply say something funny. Given this contradictory nature, sarcasm detection poses a very challenging task for NLP systems to determine, understand, and properly classify the text. One of the things that help make sarcasm detection a hard task for NLP is the fact that it is highly contextual dependent.

To advance the knowledge in NLP efforts by itself is a great reason to work on solving this challenging nuance of language, but there are many other reasons why there might be super interests in advancing and filling this gap. One of the instances, in an increasingly data driven corporate world, it is of highly important that companies be aware of what is being talked about by consumers on social media, ecommerce reviews, etc. So, the ability to detect sarcasm has tremendous value not only for the advancement of the state of the art on the matter, but also to empower companies with more and more tools to make increasingly more informed decisions. Considering these facts, the development and improvement of trained models capable of detecting sarcasm can have a tremendous impact on the society as a whole and will help us fill this peculiar nuance of natural language.

One of the best models today to understand language are the Transformers models. They understand the relationship between words in sentences, as well as detect contexts and patterns. They are highly used for classification, summarization, generations, among other NLP applications.

For this study, it was developed, fine tuned and evaluated three different transformers architectures in order to develop a comparative study of the best model to detect

sarcasm. The foundational motive is to test the inherent ability of pre-trained models on the task and set a baseline of how we can improve sarcasm detection from these architectures. The fine tuned models were RoBERTa, RoBERTa + LSTM + CNN, and GPT-2, which are detailed in the methodology section. The dataset used to fine tune the models was the News Headlines in which comprise 26,710 headlines labeled for either sarcasm or not, taken from two american news portals: The Huffington Post and The Onion, with the latter characterized by having only sarcastic headlines. The models were evaluated by assessing the accuracy, f1 score, precision and recall of the predictions.

2. Theoretical Framework

2.1. Sentiment Analysis and Text Classification

Sentiment Analysis and Text Classification are one of the most applied tasks in the realm of Natural Language Processing (NLP) [Devlin et al. 2019]. When it comes to Sentiment Analysis, it represents the ability for a model (AI) to identify - and classify - a piece of text into generally one of the following categories: positive, negative, or neutral sentiment [Pang and Lee 2008]. In other words, it helps understand the emotion and intentions behind a given textual piece. Sentiment Analysis is extremely relevant to sarcasm detection because of its contradictory nature of what is written meaning the opposite of the sentiment of the author, which imposes a challenge to understand sentiment [Gole et al. 2023]. E.g.: If it is a rainy, cloudy day outside, someone on a social media network might post: ”wow, what a day outside!”, which clearly has the potential of being a sarcastic statement. On the contrary, when it comes to Text Classification, it means literally classifying - or assigning predefined labels - to a piece of statement/text based on its own content. This task is generally a supervised learning method, in which a model is trained with a labeled dataset. This happens too with sarcasm detection. For such a task it usually involves using a labeled dataset with many instances of sarcastic and non sarcastic sentences, each with a column with the label (such as 1 or 0) and then using it to fine tune (train) a model over this dataset [Misra 2022]. The objective is to have a model that will be able to generalize to new data - which means - given a sentence it has never seen, being able to properly assign if it is sarcastic or not. So, both sentiment analysis and text classification belong to and play an important role in the challenging task of detecting sarcasm with ML and Deep Learning models. Basically, these techniques allow such detections by assessing the emotional and contextual nuances of sentences, resulting in accurate detection of the subject [Liu et al. 2019].

2.2. Transformers in NLP

In the realm of Deep Learning, it was introduced an architecture called Transformer. This architecture had an outstanding impact on Natural Language Processing by introducing a new way to process language data. Until then, the most advanced models leveraging neural networks were some such as RNNs and LSTMs [Hochreiter and Schmidhuber 1997], which performed very well processing data in a sequential way. Transformers architecture differs from them by relying on self-attention mechanisms and processing and assessing complete sentences in simultaneous and parallel ways, rather than sequentially [Vaswani et al. 2017].

One of the most significant features of Transformers is the capability of, post applying scores during self-attention mechanism in which it first understand the relative im-

portance among each of the words of a piece of text, it generates a contextual embedding, in which there's a clear understanding of the complete relevance, relation, context, and long range dependencies of each token to the whole text structure [Devlin et al. 2019]. So, it understands contextual relationships more effectively.

The gross structure of a Transformer model contains an encoder and a decoder. The encoder part is responsible for processing the input data and the decoder is responsible for generating the output, the latter usually used for tasks like text generation, translation, summarization, among others [Vaswani et al. 2017].

The transformers mechanisms and structure became the backbone of many state-of-the-art models like Bert, RoBERTa, GPT, Llama by allowing such models to handle large datasets and understand complex text patterns. So clearly, it helped improve many traditional NLP tasks, such as summarization, sentiment analysis, translation, text classification, topic modeling, etc.

2.3. RoBERTa

One of the best and most popular transformer models is BERT (Bidirectional Encoder Representation from Transformers) created by Google in 2018. BERT leverages and utilizes only the encoder part of the transformer architecture. Although BERT is on its own a large model, we utilized for the study a variation built over BERT architecture: RoBERTa (Robustly optimized BERT approach) [Liu et al. 2019].

RoBERTa differs from BERT on several aspects. The main are the fact it was trained over a larger dataset and had an extending training duration. Besides, unlike BERT, RoBERTa discarded a traditional BERT task of Next Sentence Predictions, thus focusing only on masked language modeling [Liu et al. 2019]. Masked modeling is the training by randomly masking some tokens in a sentence and forcing the model to predict what token is missing by relying on the contextual information. This fact has been shown to significantly increase the performance of the model. In addition to that, it was trained with dynamic masking, in which the masking patterns change during the training phase, so it ends up providing the model with a more robust context learning. These main differences help make RoBERTa a superior performer on typical NLP tasks, thus achieving good accuracy and generalization capabilities.

2.4. RoBERTa+LSTM+CNN

A model leveraging transformers with LSTM layers and CNN layers can provide more robustness to the task of detecting sarcasm by applying sequential and hierarchical feature extraction. It processes the text in widely different ways, which helps break and analyze it from many interesting angles [Chung et al. 2014].

While RoBERTa greatly captures contextual information and word dependencies through bidirectional attention, LSTM (Long Short Term Memory) provides the model with the ability of detecting long term dependencies and any pattern that is sequential in the text it analyzes. It belongs to RNN (Recurrent Neural Networks) architecture, with the improvement to "not forget" distant data and prevent the so-called vanishing gradient typically occurring in RNN neural networks. This makes sense for the sarcastic nature of relying on nuanced sequential information. CNN plays an interesting part by providing the model with the ability to detect patterns in text sentences. These patterns can be

from many different levels, from subtle cues among words to any possible nuances that, somehow, and not so intuitively, might indicate sarcasm [Kim 2014].

Finally, the combination enhances the model's ability to detect subtle and complex sarcasm patterns that may not be captured by RoBERTa alone.

2.5. GPT-2

The GPT (Generative Pre-trained Transformer) family models were developed by OpenAI. For the study, GPT-2 was used. GPT is a transformer architecture model designed and aimed at generating relevant text [Radford et al. 2019]. It is architectured with a large number of transformer blocks, each contemplating the previously discussed self-attention mechanisms and a multilayer perceptron (a feed-forward neural network).

Unlike BERT, the GPT models only use the decoder part of transformer architecture. Thus, it is super capable of generating text and it does so by predicting the next word of the sentence. GPT is unidirectional, understanding text only from left to right, and of an autoregressive nature, which means that it looks at the last sentence, to then generate the following. Thus, it is very good at capturing next tokens given a previous token, but performs less on understanding context compared to BERT models [Devlin et al. 2019].

However the GPTs models' primary focus is on text generation, such models can also be used for classification tasks by fine-tuning them on labeled datasets. While being fine tuned with dataset it learns to relate input texts with specific labels, effectively using and applying its whole and large pre-trained knowledge for tasks such as sarcasm detection, sentiment analysis, text classification. This fact makes GPT-2 a flexible tool for many relevant and recurrent NLP applications [Radford et al. 2019].

2.6. LoRA (Low-Rank Adaptation)

Large Language Models were trained with hundreds of millions and billions of parameters (weights). Once the training is completed, any retraining and re-sizing of its weights require a significant amount of computational capacity. One of the cases is the need to fine tune such models on new data. This is when LoRA comes in handy [Hu et al. 2021].

LoRA (Low-Rank Adaptation) is a technique that permits, rather than updating the totality of weights (parameters) of a pre-trained model during fine-tuning, it applies low-rank matrices to approximate the updates. The method consists of inserting trainable low-rank matrices to given layers of the model and at the same time keeping the first/original weights frozen.

Basically, what it does and one of its main benefits is the capacity to greatly reduce the number of weights that need to be updated, thus resulting in a fine-tuning process much more computational-efficient. As a result it is obtained faster training times and lower memory usage, which is extremely beneficial when one has limited computational resources.

Thus, LoRA effectively finds a fine line between performance and efficiency by offering the possibility to use Large Language Models in a more scalable way so it can be applied to tasks where one can benefit from model fine tuning on specific datasets.

3. Related Work

A variety of transformer models have been used in recent research to investigate sarcasm detection. [Helal et al. 2024], for example, used a contextual-based method to identify sarcasm, proving the effectiveness of transformer models in this field. In a similar line, [Misra 2022] used transformer-based techniques to find sarcasm in news headlines during their investigation. They used the same dataset used in this study. Furthermore, [Gole et al. 2023] explored the potential of GPT architectures in detecting sarcastic content by analyzing sarcasm detection using OpenAI GPT-based models. Finally, [Benavides and Ramos 2023] demonstrated the adaptability and potency of context-based models in this domain by using BERT and the CASCADE model to identify sarcasm. Together, these papers expand on our knowledge of sarcasm detection by using several transformer and contextual techniques and have served as inspiration and reference to develop this study.

4. Methodology

4.1. Dataset

For the study, we used the News Headlines, extracted from Kaggle. This dataset is made of online news headlines obtained from two american news portals: "The Huffington Post" and "The Onion". "The Huffington Post" publishes serious, formal, non-sarcastic headlines, while "The Onion" is renowned for its satirical approaches to news, with the characteristic of every headline containing sarcasm. It is on this link: `https://www.kaggle.com/datasets/rmisra/news-headlines-dataset-for-sarcasm-detection`. The dataset attributes and values are as follow:

Attributes	**Values**
Total Headlines	26,710
Columns	"headline", "is_sarcastic"
Label	1 for sarcasm, 0 for non-sarcasm
Training Set Size	21,367 (80% of total dataset)
Non-sarcastic (Train)	11,973
Sarcastic (Train)	9,394
Test Set Size	5,342 (20% of total dataset)
Non-sarcastic (Test)	3,012
Sarcastic (Test)	2,330
Data Split Method	80/20 using `train_test_split`
Balance	Well balanced for training and prediction

Table 1. Dataset Attributes

This dataset was considered very proper to kickstart the sarcasm detection study. Since the headlines were extracted from online publishing, the headlines are well written, with formal words, almost zero room for ambiguity and well constructed sentences. This eliminates the concern of models not understanding some words (what could be the case if using comments from social media where slangs, abbreviations, and misspelling are common) and allowing us to explore content, architectures and hyperparameters, and not language itself.

This original dataset also came with an additional row name "article_link", which for each headline one could access the complete news and use some additional context if necessary. Since sarcasm by its nature is highly context dependent, adding additional context makes super sense, but it skews from the purpose of this study, which is to test the inherent ability of models to detect sarcasm with supervised training using only headlines and its label.

4.2. Models

4.2.1. RoBERTa (Robustly Optimized Bidirectional Encoder Representations approach)

The RoBERTa model is specifically trained and fine tuned for text classification tasks. Its core aspect is the ability to understand the whole context of text pieces given its bidirectionality approach. It uses 12 encoder layers to process the text. I fine tuned the model using the news headlines. The train and test data is passed through the RoBERTa Tokenizer (from Hugging Face library). This tokenizer prepares the data to input into the model. It converts the tokens into number representations in a way the model can understand. It was defined "padding" as a parameter as well, aimed at guaranteeing that all the inputs have the same length, so it fills the sentences with numbers (before or after) to make it all equal in length. Also, we set "Truncation" as True to cut long sentences that would extrapolate our length definitions. These two parameters ensure equal dimensions for all sentences.

Since the task is sarcasm detection, a classification head was added on top of the RoBERTa encoder layers to enable the model to output predictions for each input sequence. The training was conducted using 5 epochs, making the model see the whole train dataset five times, and it was defined a batch_size of 8. So, backpropagation would be executed every 8 sentences passed in the forward pass. The learning rate was 2e-5 to guarantee slow and smooth optimizations during the fine tuning. The optimizer was AdamW and the loss function was CrossEntropyLoss.

To alleviate computational demands it was used Low Rank Adaptation (LoRA) to accelerate the training. LoRA applies matrix multiplication to lower the original model complexity. The R (part of the matrix) defined was 8. After defining it, we wrap it around the RoBERTa model, so now the model is more computational efficient, without losing the good attributes from RoBERTa. It is seemingly a good trade-off.

4.2.2. RoBERTa + LSTM (Long Short Term Memory) + CNN + FCNN

For this experiment, we used the same RoBERTa model architecture, with the same hyperparameters as the model in 4.1.1 with additional neural network layers. The output from the RoBERTa model is the so-called last_hidden_state. The model generates as output the contextualized embeddings vectors from the input sequences with 768 dimensions. These are high dimensional contextual embedding for each token, which capture complex semantic information and contextual information from each token and how they are related to the sentences. This contextual embedding is then fed to subsequent neural network layers (LSTM and CNN).

First, the RoBERTa output is fed into one LSTM layer with 256 neurons. The LSTM layer is responsible for detecting and analyzing possible sequential features that might exist in sarcastic sentences, because analyzing sequential structures is the core use of LSTM neural networks. Since it uses bidirectionality, its final output is a vector with 512 dimensions. Then the LSTM layer feeds a Convolution layer. It receives the 512 dim vector as input and passes it to one layer with 128 neuros (or feature maps, in the case of CNN networks) [Lecun et al. 2000]. The benefit of passing

the data through a CNN layer is that it is very effective in identifying hierarchical structures in the sequences, which transformers and LSTM might not have the same level of ability. After passing the data to a max pooling layer to reduce dimensionality and extract main and most relevant features, the whole data is embedded in a flatten layer.

Finally, the flatten layer by CNN is input into a traditional fully connected neural network with one layer of 128 neuros, which takes the data to a final, output layer with 2 neuros, each representing a class. In this case, sarcasm or non-sarcasm. Where logits will then be converted into probabilities using pytorch and we will find out to which class the sentence belongs to [Paszke et al. 2019].

4.2.3. GPT-2 (Generative Pre-Trained Transformer)

GPT2 is specially designed for generative tasks. It leverages the decoder part of the transformer architecture. For being a transformer, it also creates a contextual embedding for the text input, but since it is unidirectional, it doesn't capture the context as much as BERT models do. For the experiment, we used GPT2 with the same hyperparameters as RoBERTa: 5 epochs, 8 batch size, learning rate 2e-5, and AdamW optimizer. The model was also wrapped by LoRA to alleviate computational requirements.

4.3. Training resources

The training of all the models was done using Google Collab's T4 GPUs. Each took around 15 minutes to train.

4.4. Evaluation Metrics

To evaluate the models, we used two main metrics: Accuracy and F1-Score [Powers 2020]. Accuracy tells us a gross result of how many sentences it was corrected classified and F1score lets us understand the relationship between Precision and Recall [Sasaki 2007], which is, considering True/False Negatives, True/False Positives, and each Real Label, we would have an idea of how it exactly performed. To obtain more insights, we generated a Confusion Matrix for each model to visualize the relationship between predictions and true labels. These two metrics are conventionally used in studies with transformers when assessing the models' ability [Sasaki 2007].

$$\text{Precision} = \frac{TP}{TP + FP} \quad \text{Recall} = \frac{TP}{TP + FN} \tag{1}$$

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2}$$

5. Results and Discussion

5.1. Quantitative Results

After concluding the experiments, it was clear that the RoBERTa + LSTM + CNN model was the best one in all main metrics: Accuracy (0.9047), correcting classifying more than 90% of the sentences from the test, and F1 score (0.8827). This shows the model had a better generalization capacity on unseen data over the other ones. In second place came RoBERTa, and lastly GPT-2. The results of Accuracy and F1 score for each model tested are as follow:

A bars visualization of the performance of the models:

	RoBERTa	*RoBERTa + LSTM + CNN*	*GPT-2*
Accuracy	0.8903	**0.9047**	0.8523
F1-Score	0.8681	**0.8827**	0.8112

Table 2. Comparison of Model Performance

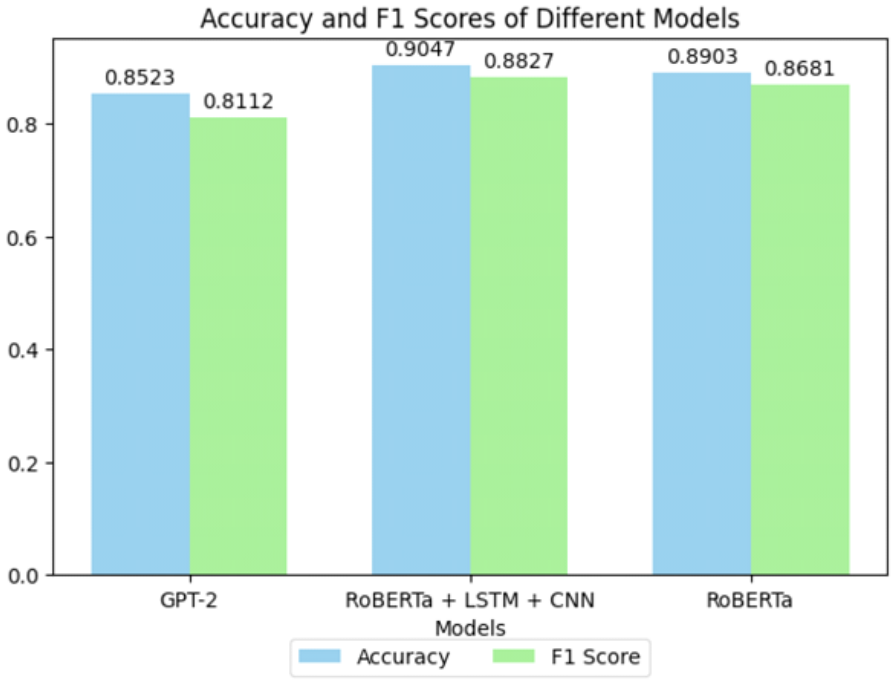

Figure 1. Accuracy and F1 Scores of Different Models

5.2. Qualitative Analysis

From this part of the analysis on, it will only be analyzed the top RoBERTa + LSTM + CNN model and GPT-2, to better understand the main differences between using a encoder and a decoder to detect sarcasm.

As noticed in the Quantitative Analysis, the best result was achieved by the RoBERTa + LSTM + CNN model, and in last place GPT-2 model.

There are many reasons why the study has shown these results. Bert models (from which RoBERTa was built upon) leverage the encoder part of the transformers, in which data pass through many encoder layers where self-attention is applied. During this process, the RoBERTa model, for being bidirectional, looks both to the right and left of each word, for every word of the input, in order to fully grasp the surroundings and context [Liu et al. 2019]. Thus, by conducting this process, this model can understand at a deep level the context of the input (in the case of the experience, the sarcastic or non sarcastic sentences), becoming naturally a good model to execute classifying tasks in NLP.

On the other hand, one of the main explanations as to why GPT-2 was the worst performer can be connected to its generative nature. For being a transformer, GPT-2 also grasps context because it generates a contextual embedding as well, but, since the model is unidirectional, it only looks at the word to the right, not understanding the context in the same level RoBERTa does. Its autoregressive nature, focused on predicting the next token, limits its ability to capture contextual

nuances in the input [Radford et al. 2019]. It also did a reasonable job when tested with unseen data after the fine tuning, but way less precise than RoBERTa.

It is also worth noting what type of sentences within sarcasm detection each model has better understand and thus can be better suited to. Regarding RoBERTa + LSTM + CNN, here's a few sentences representing its false positive and false negative predictions:

False Positives (Sarcastic predictions that were actually Non-Sarcastic)	**False Negatives (Non-Sarcastic predictions that were actually Sarcastic)**
actual soccer team loses game 46-0	police repeatedly shoot tim cook after mistaking iphone for gun
son possibly made withdrawal with dead mom	smithsonian rejects tie dylan mcdermott wore in 'the practice'
afi docs fest wraps up	south carolina refuses to remove confederate flag from capitol trailer
'got' season finale hints at appearance of that one big character	calcutta fire marshal: many indian homes lack bride extinguisher
title ix administrators discuss emotional demands of job	steven spielberg: can his career be salvaged?

Table 3. Examples of False Positives and False Negatives

Now false positives and false negatives from GPT-2:

False Positives (Sarcastic predictions that were actually Non-Sarcastic)	**False Negatives (Non-Sarcastic predictions that were actually Sarcastic)**
passport robot tells man of asian descent his eyes are too closed	nasa to send earth into space
dakota johnson awkwardly accepts sex toys from ellen degeneres	asian tsunami, hurricane katrina, kashmir earthquake battle for natural disaster award
college student wants people to eat chick-fil-a and ketchup off her body	legislators still concerned about key non-issues
'dentist offers to buy back halloween candy	report: 750,000 americans die each year during first attempt to get back in shape
once homeless student who worked 4 jobs to support family graduates college	'to defeat them, i must become them,' john kerry says while putting on black face mask

Table 4. Examples of False Positives and False Negatives (Additional)

After analyzing the sentences that were true/false positive and true/false negative on **RoBERTa + LSTM + CNN** and **GPT2** predictions, it was noticed some patterns: The **GPT-2** model showed a better performance when dealing with sentences that had sarcasm in an overly exaggerated level, correctly classifying them. On the other hand, it struggled with text that has neutral or subtle sarcasm, often not correctly interpreting sentences with true content.

The **RoBERTa + LSTM + CNN** model did an amazing job at detecting sentences with nuanced and content-specific sarcasm, being able to identify subtle cues where sarcasm was present. However, it showed bad performance classifying sentences that were true, mainly those that had

in it events and public figures names. The model excels with nuanced and context-specific sarcasm, detecting subtle sarcasm better, but it may misclassify straightforward or factual sentences, especially those involving prominent figures or events.

To exemplify a bit of the findings, it was noticed the following about the models:

Regarding GPT-2 strengths, it showed that overly exaggerated sarcastic sentences are almost perfectly identified by the model. Example: "Oh, great, another Monday! Just what I needed." This is clearly a sarcastic sentence from the test dataset. When it comes to its weaknesses, it frequently misses and does not identify nuanced or subtle sarcasm, mainly when the overly exaggerated sentences are not stated. Example: "I just love being stuck in traffic for hours!" This showed to be too subtle for the model to identify.

Concerning RoBERTa + LSTM + CNN Strengths, it was clear it deals with subtle sarcasm better, mainly when a deeper understanding of context of the input or additional knowledge is required. Example: "I'm thrilled to be working with this outdated software." It is able to spot the sarcasm in this sentence if it understands the context of what outdated software means. On the other hand, its main weaknesses are the fact that they wrongly classify some true sentences (non-sarcastic) due to slight overfitting to some common patterns. Example: "The city is planning to upgrade the public transportation system." If this is true and a real fact, that model could mistakenly classify it as a sarcastic sentence

6. Conclusion

Clearly, after conducting the experiment, RoBERTa + LSTM + CNN was the best model to use for sarcasm detection, mainly due to its bidirectional, encoder nature, in which it can grasp context and capture subtle nuances better and deeper than GPT-2. GPT-2 performed the worst, mostly due to its unidirectional, decoder architecture, which prohibits it to grasp and embrace full context of the input. GPT-2 better detected overly exaggerated sarcastic sentences and not subtle cues, which RoBERTa better spotted. Neither of the models are fully reliable to detect sarcasm by themselves, without additional context or a larger fine tuning, although they indeed can provide valuable contributions on such a task. It is important to note that there's a possibility that every model in this study could perform better if LoRA was not used, thus the models would use the full capacity of the whole parameters of each model. So, maybe there was a trade-off between accuracy and computational efficiency. For future work it is suggested that experiments be made adding more context to the sentences that the models will be trained to then predict. Also, it is worth testing with a large variety of transformers and combinations, both from BERT family and GPT, as well as exploring how adding different neural networks to the transformers can potentialize the predictions' capacity of the models.

References

Benavides, J. A. and Ramos, J. (2023). Sarcasm identification using bert and the cascade model. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING)*.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Gole, M., Nwadiugwu, W.-P., and Miranskyy, A. (2023). On sarcasm detection with openai gpt-based models.

Helal, N., Abdelgawad, A., Badr, N., and Afify, Y. (2024). A contextual-based approach for sarcasm detection. *Scientific Reports*, 14.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models.

Kim, Y. (2014). Convolutional neural networks for sentence classification.

Lecun, Y., Bottou, L., Orr, G., and Müller, K.-R. (2000). Efficient backprop.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach.

Misra, R. (2022). News headlines dataset for sarcasm detection.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library.

Powers, D. M. W. (2020). Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1.

Sasaki, Y. (2007). The truth of the f-measure. *Teach Tutor Mater*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need.

Appendices

A. GPT-2 model

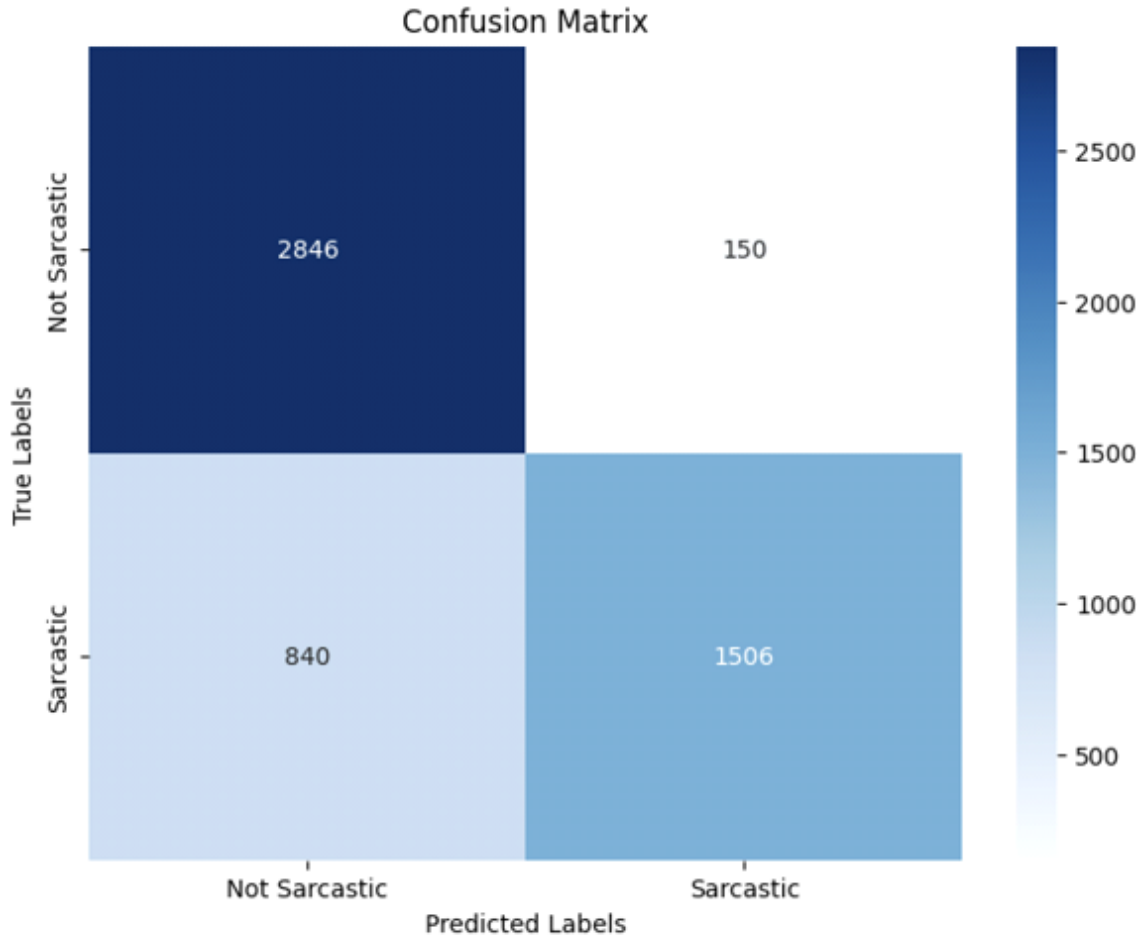

Figure 2. Confusion Matrix for GPT-2

Classification Metrics			
Class	**Precision**	**Recall**	**F1 Score**
Class 0	0.7721	0.9499	0.8518
Class 1	0.9094	0.6419	0.7526
Overall	-	-	0.8083

Table 5. Classification Metrics for GPT-2

B. RoBERTa + LSTM + CNN model

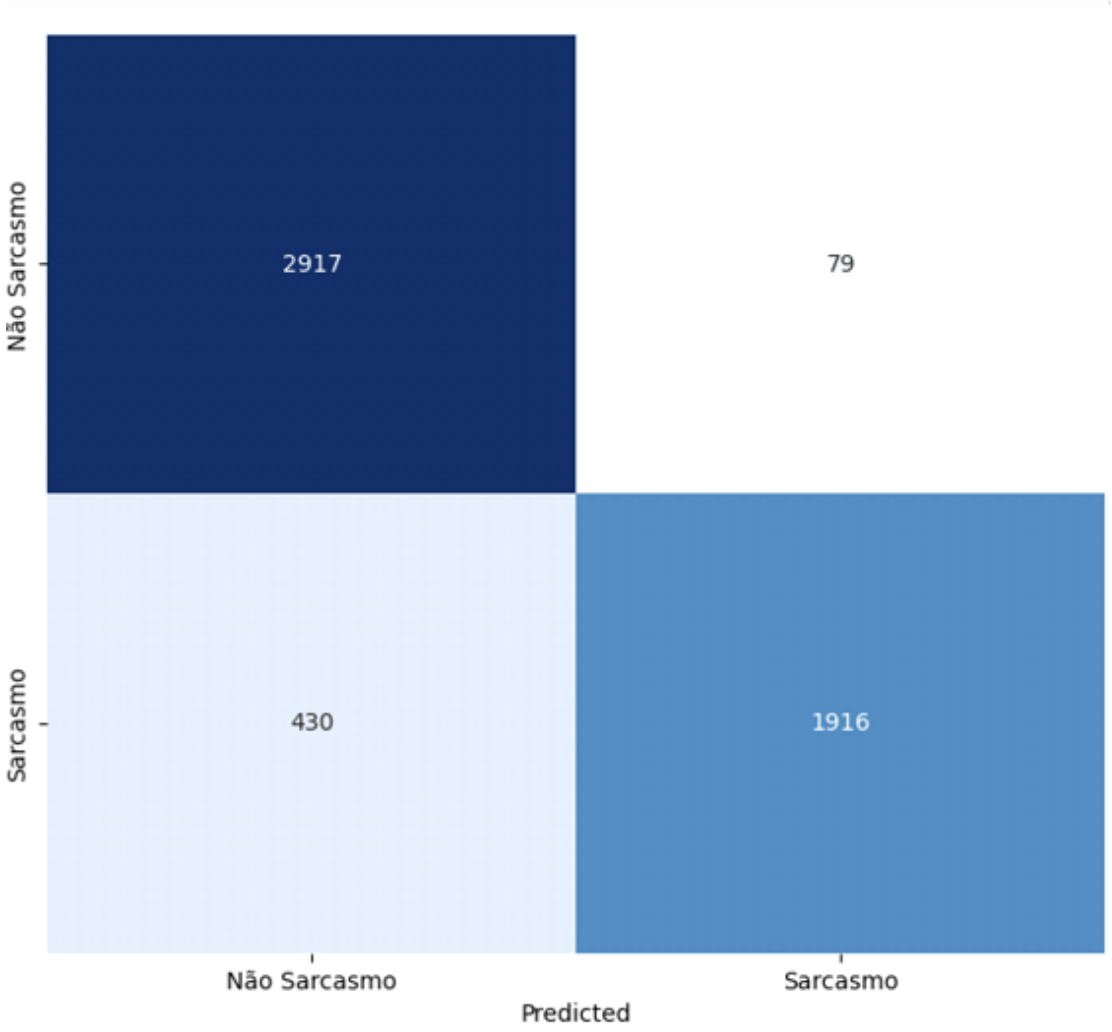

Figure 3. Confusion Matrix for RoBERTa + LSTM + CNN

Classification Metrics			
Class	**Precision**	**Recall**	**F1 Score**
Class 0	0.8668	0.9712	0.9161
Class 1	0.9568	0.8105	0.8776
Overall	-	-	0.8776

Table 6. Classification Metrics