Score-based Sentiment Classification : Comments from Reddit

Sameek Bhattacharya
University of North Texas
Denton, Texas
sameekbhattacharya@my.unt.edu

Alex Metzger
University of North Texas
Denton, Texas
alexmetzger@my.unt.edu

Venya Durgam
University of North Texas
Denton, Texas
venyadurgam@my.unt.edu

Naga Veera Subhash Alpati
University of North Texas
Denton, Texas
nagaveerasubhashalapati@my.unt.edu

Abstract

This project is intended to classify Reddit comments as positive, neutral, or negative according to their net score, defined by the number of upvotes minus downvotes. For this purpose, a total of 30,000 comments were sampled from a larger dataset, obtained from Kaggle. Three models were used for this task: a baseline using TF-IDF embeddings combined with Logistic Regression, a RoBERTa-based transformer for contextual language understanding, and a fine-tuned GPT-2 model adapted for multi-class classification. A comparative analysis of these models was performed to gain insights into which type of architecture (encoder only, decoder only, or baseline tfidf) would perform best on this dataset. The metrics used to compare model performance were accuracy, precision, recall, and F1-score, with corroborating confusion matrix analysis. There is hope to improve robustness of the models in the future by curating a cleaner dataset and adding larger feature sets to the datasets for training.

1 Introduction

In the past few years, the growth of user-generated content on platforms such as Reddit has created new opportunities for measuring public opinion and engagement. Reddit comments are particularly valuable because each one carries with it a net score that is publicly visible, indicating the difference between upvotes and downvotes from the community. In this project, we aim to categorize Reddit comments into three sentiment categories: positive, neutral, or negative, based on these net scores.

We used a publicly available dataset from Kaggle, which contains approximately one million comments across 40 subreddits on Reddit. To manage the task more conveniently, we selected a random subset of 30,000 comments for training and validation. Each record includes information such as the name of the subreddit, the comment text, a controversiality flag, and the comment score.

To classify the comments, we developed three models. The first is a simple baseline model using TF-IDF (Term Frequency-Inverse Document Frequency) embeddings combined with Logistic Regression. This basic model provided an initial benchmark for performance. The second model employed RoBERTa, a powerful pre-trained transformer capable of deep contextual language understanding, which we fine-tuned for our classification task. Finally, we adapted GPT-2, originally designed for text generation, into a multi-class classification model by modifying its tokenizer and output layers.

We evaluated the performance of these models using accuracy, precision, recall, and F1-score, along with confusion matrices to better understand classification errors. Due to the inherent bias toward neutral comments, we placed greater emphasis on precision, recall, and F1-score rather than relying solely on accuracy.

2 Related Work

Sentiment analysis has been a longstanding research area in natural language processing (NLP), traditionally focusing on categorizing text into positive, negative, or neutral sentiment based on linguistic content. Early approaches relied heavily on manual feature extraction and traditional machine learning algorithms such as Naive Bayes, Support Vector Machines (SVM), and Logistic Regression using bag-of-words or TF-IDF features (1).

With the advent of deep learning, more sophisticated methods were introduced. Word embedding models such as Word2Vec (2) and GloVe (3) enabled richer semantic representations of text, improving sentiment classification tasks. Recurrent neural networks (RNNs) and long short-term memory networks (LSTMs) became popular for capturing sequential dependencies in text (4).

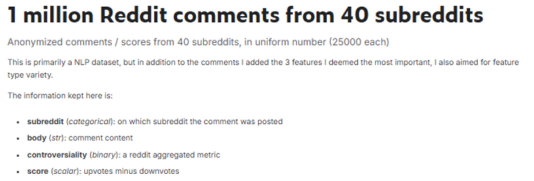
More recently, transformer-based architectures have redefined state-of-the-art performance in NLP tasks, including sentiment analysis. Models like BERT (5) and its optimized variants such as RoBERTa (6) have demonstrated superior contextual understanding through attention mechanisms, significantly outperforming earlier models. GPT-2 (7) further expanded capabilities by introducing large-scale, generative language modeling, which can be adapted for classification tasks with fine-tuning.

Specifically for Reddit data, sentiment analysis presents additional challenges due to the informal, varied, and community-driven nature of the platform. Prior works have shown that relying purely on textual features can be insufficient, motivating approaches that combine content-based models with score-driven heuristics (8).

Building on these developments, our project explores a comparative study between a traditional TF-IDF + Logistic Regression baseline and transformer-based models like RoBERTa and GPT-2, specifically fine-tuned for classifying Reddit comments into sentiment categories based on their net score.

3 Proposed Method

3.1 Dataset



1 million Reddit comments from 40 subreddits

Anonymized comments / scores from 40 subreddits, in uniform number (25000 each)

This is primarily a NLP dataset, but in addition to the comments I added the 3 features I deemed the most important, I also aimed for feature type variety.

The information kept here is:

- **subreddit** (*categorical*): on which subreddit the comment was posted
- **body** (*str*): comment content
- **controversiality** (*binary*): a reddit aggregated metric
- **score** (*scalar*): upvotes minus downvotes

Figure 1: A snap of the Kaggle page of the Dataset heading and it's description

The dataset used in this study was obtained from Kaggle and contains approximately one million Reddit comments spanning 40 different subreddits. For limitation of computational resources, a random subset of 10,000 comments was initially selected for training and validation purpose. Then 20,000 and 30,000 comments, respectively, for the transformer-based and traditional model was used for the project. Each data entry includes information such as the subreddit name, comment text, a controversiality flag, and the net score (upvotes minus downvotes). Since Reddit automatically adds an upvote when someone posts a comment, we called a comment with a score of +1 as neutral, a score that is >1 positive and <1 negative. One of the drawbacks to this approach is that it created a highly imbalanced dataset, as the vast majority of comment scores lie in between 1 and 5. This means that the positive class was present in roughly 47% of our data, the neutral class in roughly 44%, and the negative class in only 9% of the data.

Two approaches to addressing this imbalance problem were performed: stratified balanced sub-sampling and using a weighted loss function during training.

3.2 Data Preprocessing

Effective data preprocessing is crucial to ensure that the models are trained on clean, structured, and meaningful data. In this project, we applied several preprocessing techniques, including text cleaning, normalization, and tokenization where applicable.

Preprocessing steps were especially relevant in our baseline model, as there was no pre-defined implementation of the tf-idf classification model that abstracted these steps away from our code. In order to use the tf-idf vectorizer from the sklearn library effectively, we needed to use a custom vocabulary that represented our dataset better than the default vocabulary used in this package. This is due to the fact that Reddit comments use a very distinct and unusual language with very casual tones and oftentimes many mispelled words.

To build this custom vocabulary, a set of unique words was built by iterating through the 30,000 comments in our sub-sampled dataset. This generated a vocabulary of size 26,307 words.

Preprocessing the data using these methods was essential for reducing noise, standardizing the input data, and improving the performance and generalizability of our models.

3.2.1 Text Cleaning

The raw Reddit comments contained various forms of noise, such as special characters, hyperlinks, emojis, and unnecessary whitespace. To prepare the text for modeling, all comment text was first converted to lowercase to ensure consistency and eliminate redundancy arising from case differences. Special characters and extraneous punctuation were removed to simplify the input while preserving essential sentence structures. Hyperlinks, often irrelevant to the sentiment content, were detected and removed entirely. Additionally, non-standard

Unicode characters, including emojis and symbols, were eliminated to maintain textual uniformity. Finally, excess spaces, tabs, and newline characters were normalized into single spaces, ensuring a clean and consistent text format. Through these cleaning steps, we ensured that the comments provided only relevant and coherent linguistic information to the models, leading to more accurate and reliable classification results.

3.2.2 Stratified Balanced Sub-Sampling

In order to build a dataset with a sufficient number of comments belonging to each class, this technique was used to gather an equal number of entries from each class. This technique is slightly different from upsampling the minority class and/or downsampling the majority class. It works by first deciding how many of each class we would like the dataset to consist of, in this case, 10,000. Then we simply split the dataset based on class, and sample the number of each class from these sub-datasets. Lastly, we concatenated these subsets to form our dataset used for training and testing. This creates a subset of data that is perfectly balanced by class.

4 Experiments

In this study, we developed and evaluated three different models for classifying Reddit comments into sentiment categories (positive, neutral, or negative) based on their net score. Each model represented a distinct approach: a traditional machine learning baseline, a contextual transformer model, and a generative language model adapted for classification. Below, we provide a detailed explanation of each model.

4.1 TF-IDF + Logistic Regression

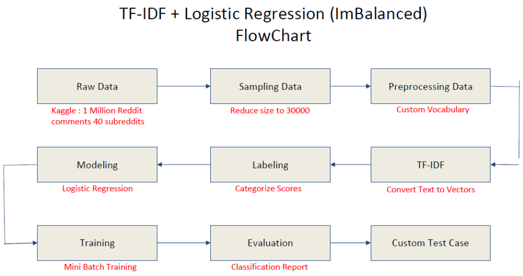


Figure 2: Pipeline followed for TF-IDF + Logistic Regression model on imbalanced data

The first model we implemented was a baseline combining Term Frequency-Inverse Document Frequency (TF-IDF) embeddings with a Logistic Regression classifier. TF-IDF transforms the raw text data into numerical vectors by considering the importance of each word relative to its frequency across all comments. After generating the TF-IDF features, a Logistic Regression model was trained to classify the comments into one of the three sentiment categories. This approach served as a simple yet effective benchmark for comparison with more advanced deep learning models.

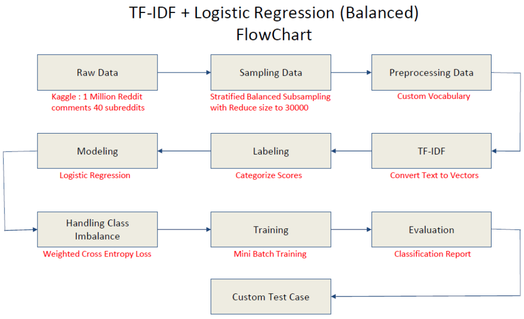


Figure 3: Pipeline followed for TF-IDF + Logistic Regression model on balanced data

The TF-IDF architecture showed several constraints both on computational efficiency and its ability to capture the deep contextual semantics present in natural language. The most resource intensive component of this model comes with the need to hold the incredibly sparse tf-idf matrix. Although the implementation of the vecotrizer used to create this matrix had optimizations in the way it was able to hold the very large matrix, which consisted of a 30,000 (comments) X 26,307 (vocabulary size) matrix, this was still the limiting factor in the number of comments we could use in our dataset before runing into OOM errors.

We trained this baseline model on both the balanced dataset generated from stratified balanced sub-sampling and the imbalanced dataset. The imbalanced dataset used the weighted cross-entropy loss function to account for the class imbalances. This required the inverse of the weights of each class to be applied to the calculated losses in order to assign a higher importance to underrepresented classes and lower importance to overrepresented classes. This showed to help the performance of the model to mimic the results seen in training the model with a regular cross entropy loss function on the balanced dataset.

4.2 RoBERTa (6)

The second model employed was RoBERTa (Robustly Optimized BERT Pretraining Approach), a

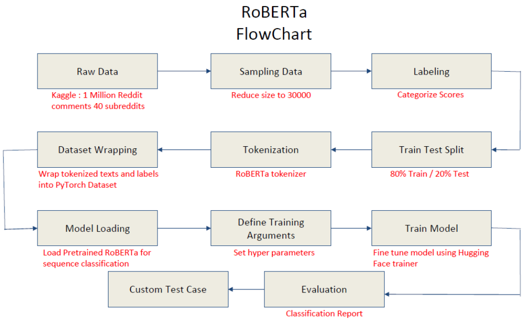


Figure 4: Pipeline followed for RoBERTa model

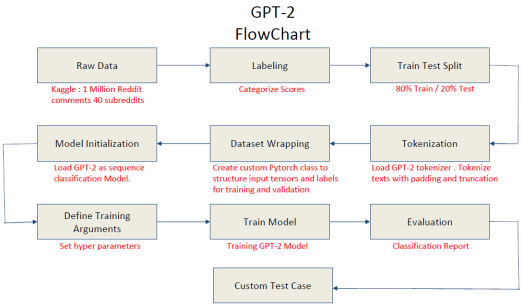

GPT-2
FlowChart
Raw Data
Kaggle : 1 Million Reddit comments 40 subreddits
Labeling
Categorize Scores
Train Test Split
80% Train / 20% Test
Model Initialization
Load GPT-2 as sequence classification Model.
Dataset Wrapping
Create custom Pytorch class to structure input tensors and labels for training and validation
Tokenization
Load GPT-2 tokenizer . Tokenize texts with padding and truncation
Define Training Arguments
Set hyper parameters
Train Model
Training GPT-2 Model
Evaluation
Classification Report
Custom Test Case

Figure 5: Pipeline follwed for GPT-2 model

transformer-based model known for its strong performance in natural language understanding tasks. RoBERTa is an encoder-only model that builds upon BERT by using dynamic masking, training on larger datasets, and removing the next-sentence prediction objective. In our project, RoBERTa was fine-tuned on the cleaned Reddit comments, with a classification head added to output probabilities across the three sentiment classes. The input comments were tokenized using the RoBERTa tokenizer to ensure compatibility with the model's pre-trained vocabulary. RoBERTa demonstrated a significant improvement in classification accuracy and F1-score, highlighting its ability to capture nuanced patterns and context within Reddit discussions.

During training, RoBERTa was leveraged to encode each comment into a contextualized representation that captures syntactic structure and semantic nuance, allowing the classification head to distinguish between positive, negative, and neutral score connotations more effectively. RoBERTa's biggest improvement over the other architectures used in this project is its bidirectional attention mechanism, which enables it to consider the full context of each word within the sentence. This concept dramatically improves the models capability for interpreting slang, sarcasm, or mispelled wording and casual language often found in Reddit comments. RoBERTa demonstrated some improvement in both classification accuracy and macro-averaged F1-score over the simpler TF-IDF baseline, likely due to its ability to capture subtle patterns in user-generated content.

4.3 GPT-2 (7)

The third model we utilized was GPT-2, originally designed for generative tasks, and was adapted here for multi-class classification. GPT-2 is a decoder only model whose sole purpose is to predict the next token given a string of previously seen tokens. This changes the attention mechanism used from the biderectional one previously discussed in RoBERTa to a unidirectional attention mechanism. That is, GPT-2 relies soleley on previous tokens when building context of each word and has no notion of words/tokens that appear later in the document.

Fine-tuning involved modifying GPT-2's output layer to predict class probabilities rather than generating text sequences. Additionally, special care was taken in preprocessing, including adjusting the tokenizer to handle padding appropriately for batch training. Despite being a generative model, GPT-2 exhibited strong classification performance, showcasing its versatility. However, it occasionally misclassified comments near the boundary between neutral and positive sentiments.

5 Results and Discussion

In evaluating the performance of the 3 models, the transformer based models showed improvements on the baseline TF-IDF + Regression architecture. RoBERTa and GPT-2 demonstrated comparable performance in terms of accuracy and macro-averaged F1-score, indicating that both encoder-only and decoder-only transformer architectures are potentially effective for classifying the Reddit comments for this task, although more work needs to be done.

The results do offer insights on a comparative analysis between the 3 models used on this dataset, however, the the fairly low f1 scores, particularly for the minority (negative) class dictate more needs to be done in order to predict the scores in a sufficiently balanced manner.

5.1 Analysis

The precision, recall, and f1 scores are given in table 1 below. These results indicate that the only

Table 1: Precision, Recall, and F1-scores for the 3 models analyzed in the project

Model	Class	Precision	Recall	F1-Score
TF-IDF + Logistic Regression (balanced)	Negative	0.41	0.41	0.41
	Neutral	0.43	0.35	0.39
	Positive	0.38	0.43	0.40
TF-IDF + Logistic Regression (imbalanced)	Negative	0.10	0.12	0.11
	Neutral	0.45	0.45	0.45
	Positive	0.55	0.54	0.54
RoBERTa (fine-tuned)	Negative	0.27	0.07	0.11
	Neutral	0.54	0.41	0.47
	Positive	0.58	0.75	0.65
GPT-2 (fine-tuned)	Negative	0.20	0.01	0.01
	Neutral	0.54	0.45	0.49
	Positive	0.58	0.76	0.65

effective way we found to predict the minority class was to balance the dataset manually using stratified balanced sampling. Even the better performing transformer models struggled with this class when the imbalanced dataset was used, despite their better overall f1 metrics.

When examining table 1, the most valuable insight comes from evaluating the other two classes, as this was the bulk of the dataset. As the comparative analysis, stands, both the encoder-only (RoBERTa) and decoder-only (GPT-2) performed similarly on the imbalanced dataset, both of which show significant improvement over the TF-IDF + Logistic Regression model in terms of highest class f1-score.

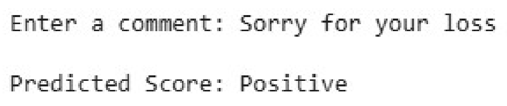
5.2 Inference

All 3 models used the same test cases to perform inference on in order to get a meaningful example of score prediction across the different models that were trained during this experiment. When creating these test comments, it was important to keep in mind that this was not strictly a sentiment analysis task for these test cases. That is, the models were not trained to place the test comments into a positive, neutral, or negative class based on their contextual sentiment, but rather predict how many net upvotes/downvotes the comment would have. For this reason a test set such as:

Negative: *This movie was terrible*
Neutral: *This movie was ok*
Positive: *This movie was fantastic*

does not give a valuable representation of the types of comments we would expect the model to classify. Rather, we would expect the model to have an outstanding ability to classify more devisive and influential comments such as:

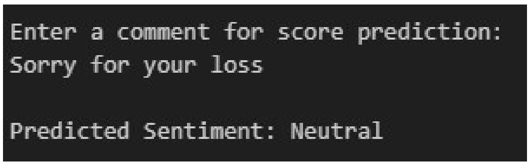
Negative: *You are an awful person and I hope bad things happen to you to make you more miserable*
Neutral: *Sorry for your loss.*
Positive: *So sorry for your loss! He was lucky to have you and I'm sure you meant more to him than you know.*



```
Enter a comment: Sorry for your loss

Predicted Score: Positive
```

Figure 6: An example of score-based sentiment prediction : Balanced TF-IDF + LR model



```
Enter a comment for score prediction:
Sorry for your loss

Predicted Sentiment: Neutral
```

Figure 7: An example of score-based sentiment prediction : RoBERTa model

6 Conclusions

In this study, we evaluated multiple models for the task of comment classification, focusing on their performance in handling imbalanced datasets and accurately predicting minority classes. Our experiments encompassed both traditional machine learning and transformer-based models, including

TF-IDF with Logistic Regression, RoBERTa, and GPT-2.

The TF-IDF model, while relatively simple, showed meaningful gains when the dataset was balanced using stratified sampling. Specifically, the average F1-score improved from 0.36 to 0.4, highlighting the importance of addressing class imbalance in text classification tasks. This demonstrates that even baseline models can benefit significantly from careful preprocessing and sampling strategies.

Among the models tested, RoBERTa consistently outperformed others in terms of accurately classifying the minority class. Its contextual understanding of language allowed it to better differentiate between nuanced comment categories. GPT-2 also showed potential but required substantial computational resources and hyperparameter tuning to reach comparable performance, making it less practical in resource-constrained settings.

Overall, the results emphasize the impact of data balancing techniques and the power of transformer-based models in natural language processing tasks. These insights can guide future efforts in improving comment classification systems, particularly when dealing with skewed class distributions and context-sensitive content.

6.1 Future Work

A key improvement planned for future iterations is the inclusion of additional metadata such as the subreddit name and the original post title or body associated with each comment. These additional bodies of text can provide valuable background and context to the comment text that these models have been initially trained on. For instance, a comment that appears sarcastic or neutral in isolation may convey a clearly negative or positive sentiment when considered in the context of the original post.

To integrate this metadata information, future work will involve expanding the input representation to concatenate or otherwise embed both the comment and relevant contextual features. Models like RoBERTa and GPT-2 are well-suited for multi-segment input and can potentially benefit from training on this richer data format. Additionally, multi-input transformer architectures can be explored to effectively encode both comment-level and post-level features.

By enriching the dataset with subreddit and post context, we anticipate improvements in the classification of these Reddit comments. Hopefully, we can use these unexplored techniques to build on the results obtained in this experiment and use the information harnessed from this comparative analysis as a launchpad for more robust, better performing models in the future.

References

[1] B. Pang, L. Lee, and S. Vaithyanathan, *Thumbs up: Sentiment Classification using Machine Learning Techniques*, Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 79–86, 2002.

[2] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient Estimation of Word Representations in Vector Space*, arXiv preprint arXiv:1301.3781, 2013.

[3] J. Pennington, R. Socher, and C. D. Manning, *GloVe: Global Vectors for Word Representation*, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, 2014.

[4] S. Hochreiter and J. Schmidhuber, *Long Short-Term Memory*, Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] J. Devlin, M. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, arXiv preprint arXiv:1810.04805, 2018.

[6] Y. Liu et al., *RoBERTa: A Robustly Optimized BERT Pretraining Approach*, arXiv preprint arXiv:1907.11692, 2019.

[7] A. Radford et al., *Language Models are Unsupervised Multitask Learners*, OpenAI, 2019.

[8] S. Gilda, *Machine Learning for Fake News Detection*, arXiv preprint arXiv:1705.00648, 2017.