Shahid Beheshti University
Data Science (2023-2024)

Steam Review Generator Report

Mirmahan Madani
Mohammad Mehdi Begmaz

Department of Computer Science
Shahid Beheshti University

mi.madani@mail.sbu.ac.ir
mohammad.begmaz@cs.sbu.ac.ir

Abstract

The game industry receives a large amount of user-generated content, particularly in the form of reviews, which provides both a wealth of knowledge and a challenge in processing such large datasets. The study provides insight into GPT-2's ability to capture user feelings in game reviews, which has possible implications for natural language generation in the gaming industry. Using existing game reviews as training, the GPT-2 algorithm generates informative reviews that capture user sentiment. A subsequent GPT-2-based classifier classifies the generated reviews as either positive or negative.

1 Introduction

The digital era has resulted in an unparalleled quantity of user-generated content, particularly in the field of gaming, where fans share their experiences and opinions via online reviews. A huge amount of data makes it difficult to effectively exploit its insights. The current project addresses the need for automated game review production and sentiment classification, to streamline the process of extracting useful information from a huge repository of user comments.

The greatest challenge is the human work necessary to sort through and understand all of the game reviews. This operation requires a significant amount of time and resources, but it also carries the risk of ignoring minor differences and trends in the data. To address these issues, we turn to advanced language models, especially GPT-2, recognized for its natural language processing skills.

We aim to automate the development of game reviews using GPT-2, capturing the depth of

user sentiments. However, the project introduces an extra element of complexity by ensuring that the created content fits the complex language and context typical of gaming reviews.

Furthermore, we extend the scope to include sentiment classification, recognizing the subjective character of reviews. While generating content, the model must distinguish between positive and negative thoughts, reflecting the fundamental challenges of human opinion.

In establishing these challenges, our project not only includes the practical issues of managing massive amounts of user-generated data but also explores the complex interplay between language models and sentiment analysis in the gaming world. The next sections describe our methodology, approach, and conclusions, providing insight into the novel potential of using advanced language models in game reviews.

2 Related work/Background

As the connection between natural language processing and sentiment analysis becomes more visible, several recent studies have dealt with the issues of automated content generation and sentiment classification. Two significant approaches stand out for presenting solutions to similar challenges, although in different contexts.

Transformers: Cutting-edge Natural Language Processing (Wolf et al., EMNLP 2020): This significant study investigates the transformative effects of transformer-based models, such as GPT-2, in natural language processing. Transformers have transformed language models' understanding and generation of textual material by introducing attention and self-attention mechanisms. While not specifically focused on gaming reviews, this study provides an excellent understanding of the technology that supports our approach.

Researchers have employed a variety of approaches to sentiment classification (Liu, 2012; Pang & Lee, 2008). Sentiment analysis methodologies frequently rely on sentiment lexicons to figure out which words or phrases are positive or negative in a general or domain context. General Inquirer (Stone et al., 1966) is a manually generated resource that is frequently used in sentiment analysis. Many methods have been presented for determining the polarity of sentiment words (Huang et al., 2014; Lu et al., 2011; Qiu et al., 2009).

In general, there are two basic approaches to sentiment analysis: machine learning and linguistics (also known as natural language processing) (Shaikh et al., 2008). Because sentences are brief and do not contain many subjective terms, the machine learning approach frequently suffers from limited data issues. Furthermore, the machine learning approach cannot handle complicated grammatical relationships between words in a phrase. Some researchers have used linguistic features in addition to word features in machine learning to address the constraints of the bag-of-words (BOW) technique (Liu, 2012).

By considering these varied works, we receive inspiration and ideas that shape our technique and strategy. The combination of transformer-based models and sentiment analysis techniques serves as the foundation for our research, these developments to the particular challenges of automated game review generation and sentiment classification.

3 Proposed method

3.1 Preprocessing and Feature Engineering

Preprocessing the dataset involved multiple steps to determine the ideal features and select the most effective ones. Basic data analysis showed the dataset had a number of flaws; namely missing values for the ‘review’ column and a number of duplicated reviews. To prevent the model from getting overfit, all duplicate data points were removed (except the first instance) and any record with a missing value in the ‘review’ column was dropped.

Text Formatting: Cleaning and formatting the text data, also known as the corpus, is another major preprocessing task. Using Regular Expressions, the text corpus can be formatted in a way that is more appropriate for tokenization and model training. The goal is to select the words that can help us train the model as efficiently as possible. This means non-alphanumeric characters and other elements such as markdown tags should be removed. The text corpus is converted to lowercase to simplify the tokenization process.

Feature engineering: One additional feature that we added to the dataset is word_count, which is the number of words in a review. It is preferred to use text inputs that are approximately the same length as each other for fine-tuning or training a language model, so this feature can be used to select records that are more similar to each other.

Profanity check: Steam does not handle profanity in its review system, so it’s up to us to detect profane reviews and remove them. The goal is to prevent the model from learning and generating potentially offensive and inappropriate text.

Data selection: Finally, a limited collection of records must be selected for the actual training process. Records containing reviews that are too short or too long should be dropped. Additionally, records with zero number of upvotes can be dropped as they were not popular enough to grab the attention of a human.

We aim to select the most helpful reviews, so out of all the remaining records, the top 10’000 were chosen based on their weighted_vote_score. This feature is calculated by Steam as an internal measure for review popularity. Steam uses the weighted_vote_score to display the most popular reviews for each video game, so it seems logical for us to sort our data based on it.

3.1.1 Tokenization

Tokenization is a key process in natural language processing that involves dividing a given text into smaller units called tokens. These tokens provide a structured representation of the text. The primary goal of tokenization is to facilitate language processing by converting raw text into a format that can be easily understood and analyzed by computer algorithms.

A tokenizer is in charge of tokenizing the inputs for a model. The Tokenizer library contains pre-trained tokenizers for all the models used in this project. We specifically used the AutoTokenizer class which automatically creates the corresponding tokenizer object for a given model.

The tokenization process can be controlled via a number of optional arguments, including:

- **padding:** Ensures that all sequences have the same length.
- **max_length:** Specifies the maximum length of the sequences after padding or truncation.
  Sequences longer than this value will be truncated, and sequences shorter will be padded.
- **truncation:** Sequences longer than the specified max_length will be truncated. It is useful to ensure uniform sequence lengths.
- **num_proc:** Number of processes to use for tokenization in parallel. Useful for speeding up tokenization when processing a large dataset.

Padding in general is imperative for ensuring a smooth and efficient training process. During the training process, if all the text data in a batch isn't the exact same size, it must be dynamically padded. This is done via a **DataCollator**, which receives the tokenizer object as input. Our generative model uses a DataCollatorForLanguageModelin.

3.2 Model Architecture

With the aim of automated game review creation and sentiment classification, our model architecture is based on the powerful GPT-2 language model. GPT-2 is a self-supervised transformers model that was pre-trained on a huge corpus of English data. This means that it was pre-trained on raw texts only, with no human labeling (which allows it to handle a large amount of publically available data), and it utilized an automatic mechanism to generate inputs and labels from those texts. Specifically, it was trained to predict the next word in sentences.

More exactly, inputs are sequences of continuous text of a given length, while targets are the same sequence shifting one token (word or chunk of word) to the right. The model internally employs a masking mechanism to ensure that predictions for a token only use inputs from 1 to me and not future tokens.

This allows the model to develop an inner representation of the English language, which can subsequently be utilized to extract features helpful for subsequent tasks. The model excels at what it was trained for creating texts given a prompt.

For our analysis, we choose the base version of GPT-2, which has 124M parameters. GPT2ForSequenceClassification and AutoModelForCausalLM(gpt2) are two different GPT-2 variations that we used, each designed to address different parts of our task:

3.2.1 GPT2ForSequenceClassification:

GPT2ForSequenceClassification serves as a foundation for sentiment classification in our model architecture. This version of GPT-2 is optimized for sequence classification tasks, allowing it to detect feelings presented in generated game reviews. The pre-trained GPT-2 model is fine-tuned for our sentiment classification task, learning to map generated reviews to binary sentiment labels (positive or negative).

This is a high-level summary of the architecture.

- Transformer Architecture:

GPT-2 is based on the transformer architecture, which includes several layers of self-attention mechanisms and feedforward neural networks. This architecture enables the model to accurately capture contextual dependencies within sequences.

- Embedding Layer:

  The input layer converts tokens from the input sequence into high-dimensional embeddings. These embeddings encode semantic information about the words or tokens in the sequence.

- Positional Encoding:

  To fit the language's sequential nature, positional encodings are added to the embeddings, which provide information about each token's position in the sequence.

- Attention Mechanism:

  The attention mechanism is at the core of the transformer architecture, allowing the model to prioritize different parts of the input sequence while making predictions. This method enables GPT-2 to detect long-range relationships in data.

- Feedforward Neural Networks:

  Following the attention mechanism, each transformer block has feedforward neural networks that process data and contribute to model predictions.

- Pooling and Classification Layer:

  In the case of GPT2ForSequenceClassification, the model's final hidden states are combined before a classification layer is added to generate the output. This output is utilized for binary classification tasks, including sentiment analysis.

3.2.2 AutoModelForCausalLM(gpt2):

To aid the development of game reviews, we use the AutoModelForCausalLM variation, which is implemented with the GPT-2 architecture. This variation is specifically developed for language modeling problems, in which the model predicts the next token in a sequence based on the context of the previous tokens.

Using GPT-2's causal language modeling capabilities, we can generate coherent and contextually appropriate game reviews. During the fine-tuning process, the model learns to capture the underlying structure of game-related language, ensuring that the generated reviews reflect the intricate patterns and sentiments found in user-provided gaming material.

The design is similar to that of GPT2ForSequenceClassification, but it focuses on language production rather than sequence classification.

- Transformer Architecture:

  AutoModelForCausalLM(gpt2), like the sentiment classification variation, is created using the transformer architecture.

- Embedding Layer and Positional Encoding:

  Input tokens are turned into embeddings, and positional encodings are used to capture the data's sequential nature.

- Causal Masking:

  In language modeling challenges, a causal mask is used during self-attention to prevent the model from attending to future tokens. This assures that the model creates tokens one at a time, in a causal sequence.

- Decoder Architecture:

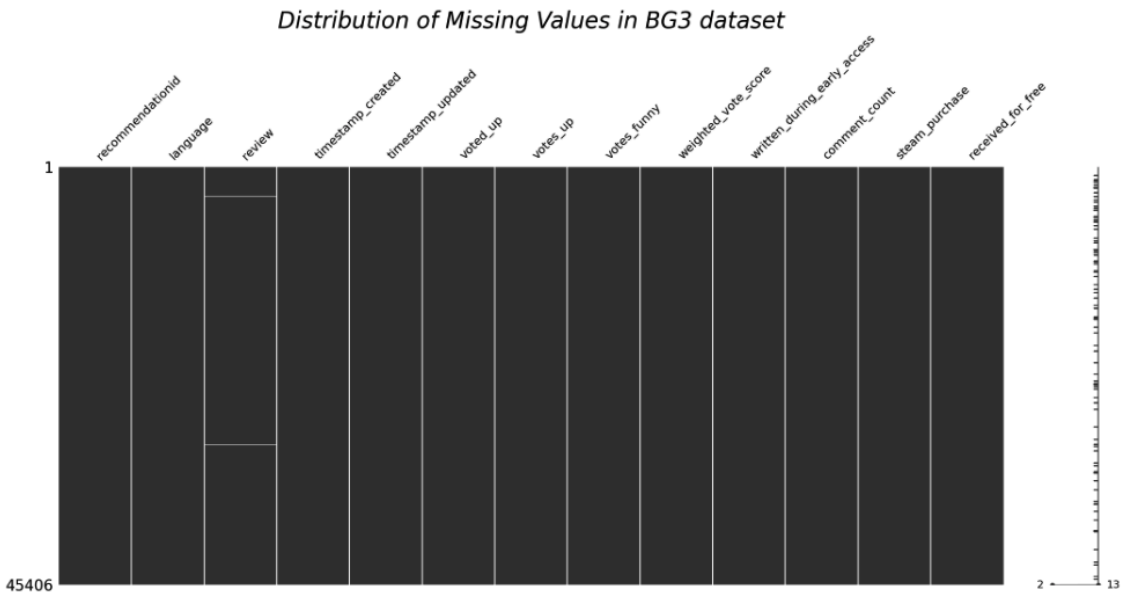
  GPT-2, as a causal language model, provides decoder-only. The decoder constructs sequences autoregressively, predicting one token at a time using the context provided by the previous tokens.

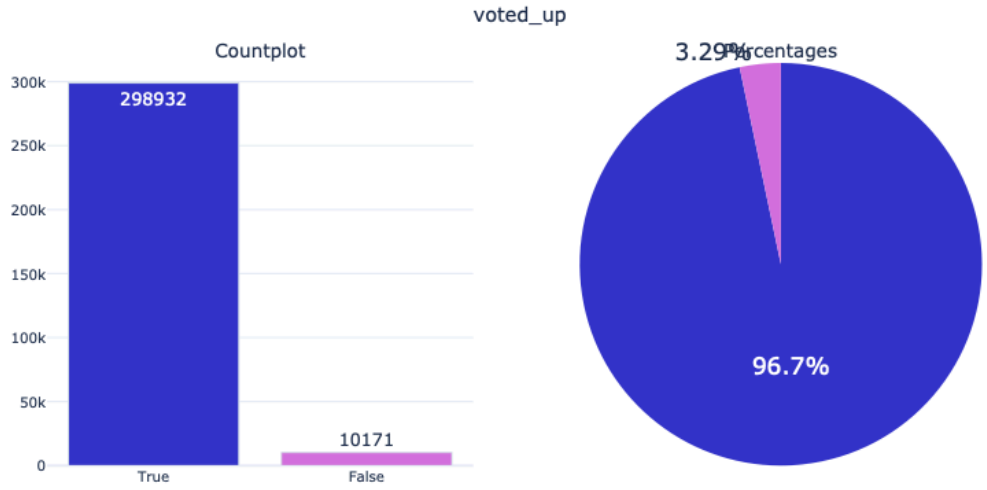
- Output Layer:

  The last layer generates a probability distribution over the vocabulary for the following token, enabling the creation of diverse and context-relevant language.

3.3 Fine-Tuning Methodology

In this project, achieving optimal performance in automated game review production and sentiment classification is dependent on an effective fine-tuning strategy. The chosen approach includes Progressive Early Fine-Tuning (Peft) with a special focus on the Layer-wise Relevance Adjustment (LoRA) technique, which improves the GPT-2 model's adaptability for our goal objective.





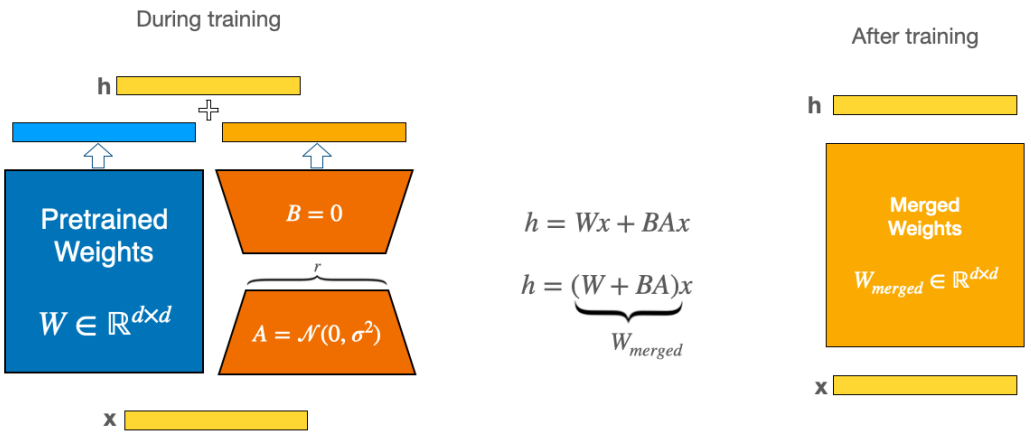
3.3.1 Progressive Early Fine-Tuning (Peft):

Peft can be utilized as a fundamental technique to fine-tune pre-trained models such as the GPT-2 model. This method uses an iterative process in which the model continues to be adapted to the demands of the gaming domain. During the initial phases of fine-tuning, the model is exposed to a wide range of gaming-related material, helping it to understand domain-specific terms, context, and sentiment variations. This gradual adaptation reduces the risk of overfitting to a specific subset of the data, hence improving the model's generalization capabilities.

3.3.2 Low-Rank Adaptation (LoRA):

To improve the fine-tuning process, we use the Layer-wise Relevance Adjustment (LoRA) strategy, which aims to speed the adaption of pre-trained models to new tasks. LoRA improves efficiency by adopting an interesting strategy: expressing weight updates with two smaller matrices using low-rank decomposition. This method enables effective adaptation to new data while reducing the overall number of parameter changes. Additionally, the original weight matrix is left unchanged, protecting the knowledge gained during pre-training, while the adaptive matrices are integrated to provide the final results.

While LoRA is significantly smaller and faster to train, the fact that the base and LoRA models are loaded independently may cause latency concerns during evaluation. To decrease delay, use the merge_and_unload() function to combine the adapter weights and the base model, allowing you to use the newly merged model as a standalone model effectively.



This works because, during training, the smaller weight matrices (A and B in the picture above) remain separate. However, once training is completed, the weights can be combined into an identical weight matrix.

The combination of Peft and LoRA allows a fine-tuning pipeline adapted to the challenges of game review creation. This methodology optimizes the GPT-2 model for our unique goal, of providing a balance of adaptability and specialization. The iterative structure of Peft, together with the targeted changes offered by LoRA, contributes to the model's ability to generate logical and contextually relevant game reviews while remaining sensitive to the many moods expressed by players.

4 Results

4.1 Dataset

The Baldur's Gate 3 Steam Reviews dataset was used extensively for this project, as a sample from the many available suitable datasets. It contains more than 300’000 records of reviews from real Steam users who have purchased the video game Baldur’s Gate 3 (2023). Note that the dataset has already been slightly processed, as it only contains reviews written in English.

These data have important features, the most important of which will be explained below:

- Review: text of written review
- Timestamp_created: date the review was created (unix timestamp)
- Voted_up: Indicates whether the review is positive or negative
- Votes_up: number of users that found this review helpful
- Votes_funny: number of users that found this review funny
- Weighted_vote_score: helpfulness score

These features used to train models for example for our classification we used review as input and voted_up as our labels to train model and for the causal model we used review as input and target just like AutoEncoder models.

To download the dataset or see more information about it, you can refer to the link below:

https://www.kaggle.com/datasets/a8a570d2cc8c92e08704f43542fcb88d91982950b1067b192516bbd50bffa422

Here is some important plots and information about this dataset:

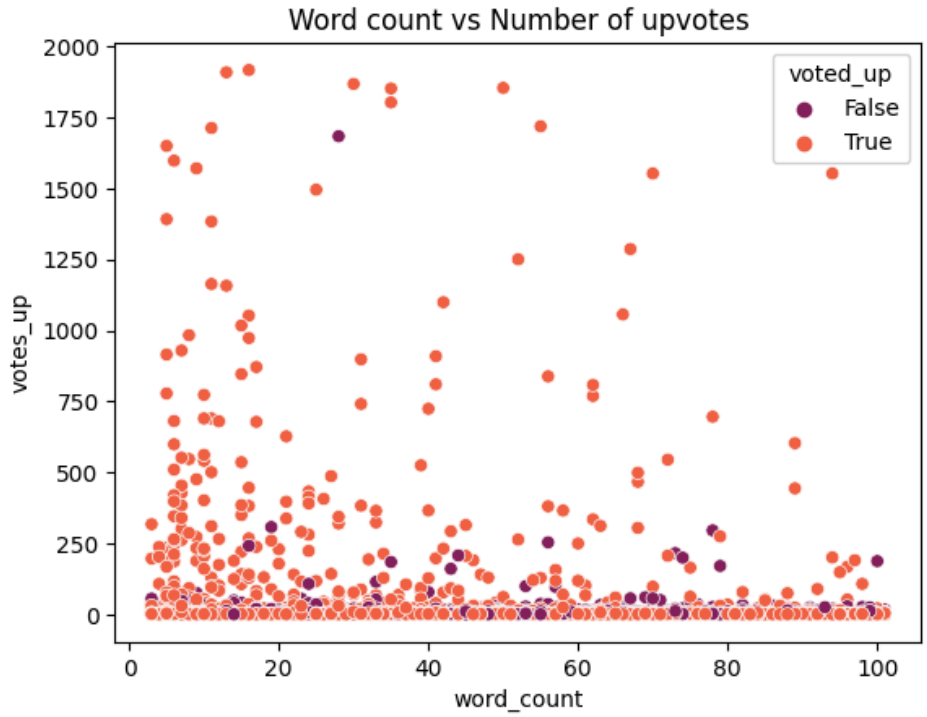
- SHAPE:
  - ROWS: 45406
  - COLS: 13

- TYPES:
  - Recommendationid → int64
  - Language → object
  - Review → object
  - Timestamp_created → float64
  - Timestamp_updated → float64
  - Voted_up → object
  - Votes_up → float64
  - Votes_funny → float64
  - Weighted_vote_score → float64
  - Written_during_early_access → object
  - Comment_count → float64
  - Steam_purchase → object
  - Received_for_free → object

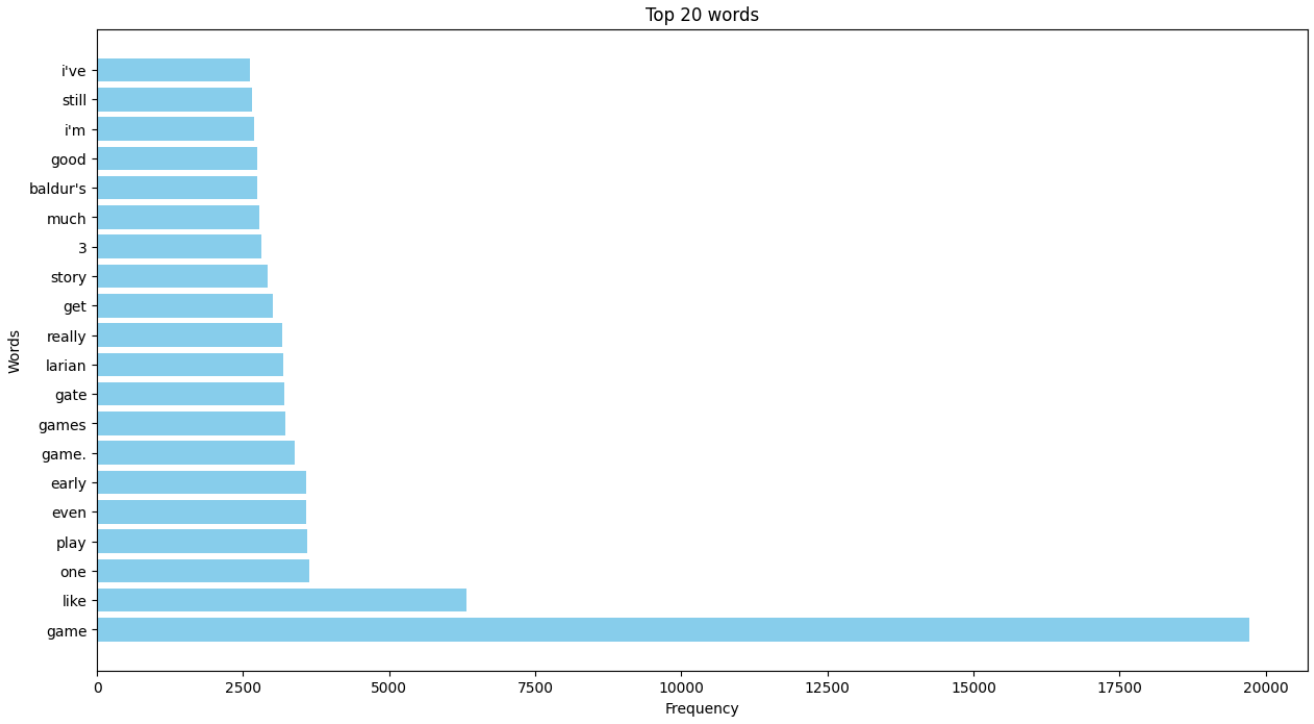
- MISSING VALUES
  - Recommendationid → 0
  - Language → 0
  - Review → 90
  - Timestamp_created → 1
  - Timestamp_updated → 1
  - Voted_up → 1
  - Votes_up → 1
  - Votes_funny → 1
  - Weighted_vote_score → 1
  - Written_during_early_access → 1
  - Comment_count → 1
  - Steam_purchase → 1
  - Received_for_free → 1
  - dtype: int64

- DUPLICATED VALUES
  - NUMBER OF DUPLICATED VALUES → 0
  - Note that fully duplicated values were not present in this dataset. However, the 'review' column contained over 60'000 identical values. The records with an identical 'review' feature were dropped to avoid overfitting the model.

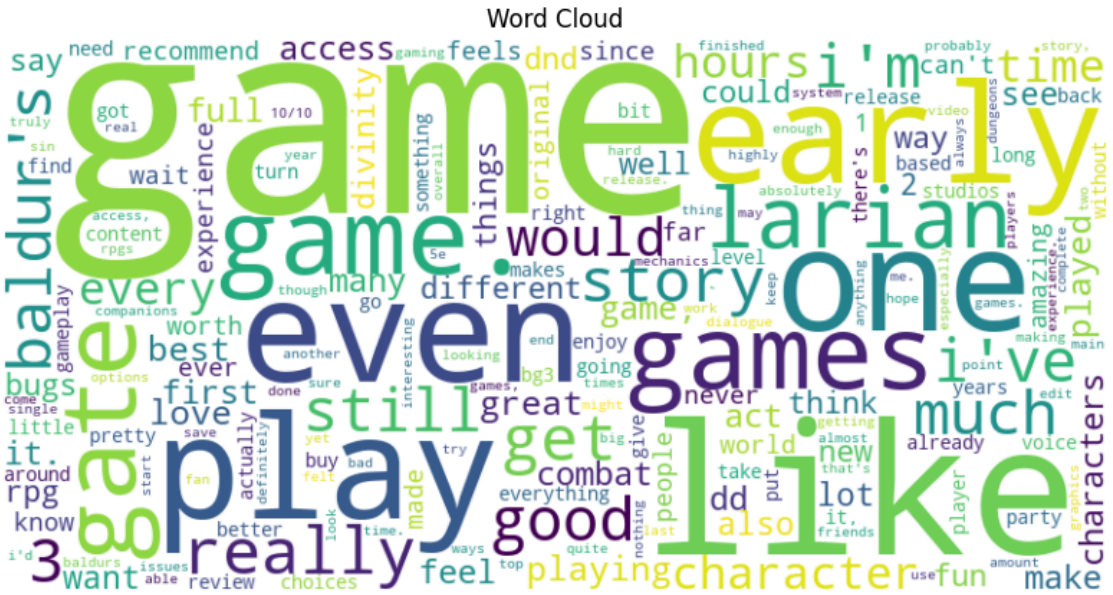
The following scatter plot displays the relationship between the number of words in a review and the number of upvotes it has received from other users. Interestingly, some of the most highly upvoted reviews have a length of less than 20 words. It appears as if increasing the word count will gradually reduce its chance of receiving upvotes.



The following plot displays the frequency of the top 20 most used words in the reviews. We can even see a word like ‘Larian’ which is the name of the studio behind Baldur’s Gate 3.



This word cloud shows the most frequented words in another view. Words like ‘game’ and ‘play’ and others are very frequent in our data so we can expect the generative to model to use them often.



4.2 Experiments

For this project, a total of 4 models were fine-tuned for Causal LLM, and 2 models were fine-tuned for Sequence Classification. The goal of the causal models is to generate text sequences that resemble the reviews written on Steam for Baldur’s Gate 3, The goal of classification models is to analyze Steam reviews (both human-written and AI-generated) and determine whether the review is meant to be positive or negative.

Note that the classification model also gained the ability to classify almost any type of text, not just Steam reviews, Opening up its use cases for more creative purposes.

All of the models trained for this project are based on GPT-2. A wrapper class was used to tokenize the input data for ease of use. All causal models were evaluated using the **ROUGE** metric and all classification models were evaluated using the **Accuracy** metric.

Below you can find a list of all of the causal models and their features:

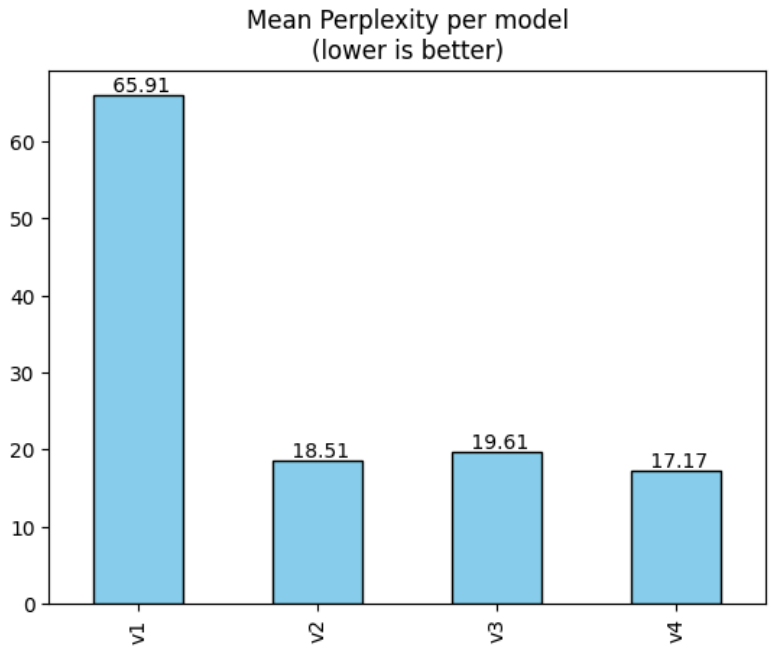
- **Causal LLM v1:** The first version of the generative model. This model is based on GPT-2 (using AutoModelForCausalLM) and was fine-tuned on 10’000 data points consisting of the most high-quality reviews in the dataset. However, the tokenization method used for it was not suitable, resulting in below-average performance, while also taking a significant amount of time.

- **Causal LLM v2**: The second iteration of the model, trained over 5 epochs. The tokenization strategy was completely changed for training this model. All of the input text of each record was concatenated to solve the padding issue. While this solution proved to be effective, the concatenation strategy created a number of other issues.
- **Causal LLM v3:** The third iteration of the model, using the same tokenization strategy as version 2, but missing the concatenation component. It was trained over 5 epochs and has generated acceptable results.
- **Causal LLM v4:** This model is the exact same as version 3, but instead of 5 epochs it was trained for 10 epochs. Based on loss values, it is slightly overfit and performs marginally worse than the previous version.

Evaluation of Causal Models:

All 4 versions of this model were evaluated using ROUGE and Perplexity.

Perplexity: Intuitively, perplexity means to be surprised. We measure how much the model is surprised by seeing new data. The lower the perplexity, the better the training is.



The mean perplexity was calculated using 10 samples generated by each model. The numbers you see above may change if the test is run again or with a larger number of samples. Regardless, based on this plot we can see a clear reduction in perplexity going from model version 1 to the other versions.

Below you can find a list of all of the classification models and their features

- **Classification model v1:** The first version of our classification model. This model is based on GPT-2 (using AutoModelForSequenceClassification) and was fine-tuned on the same 10'000 data points as the causal models. Due to the imbalance of negative and positive reviews in the dataset (with over 95 percent of the reviews being positive), this model isn't capable of classification and reports all inputs as positive.
- **Classification model v2:** The second iteration of the classification model. This model was fine-tuned on a more balanced dataset with 5000 positive and 5000 negative records. It is surprisingly effective at classifying reviews (and other text formats) as being mostly negative or mostly positive. It can potentially be used in sentiment analysis tasks.

Evaluation of Classification Models:

Both versions of this model were evaluated using accuracy, showcasing how effective they are at classifying reviews as positive or negative.

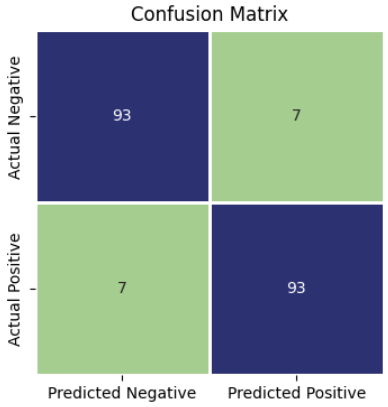
Both models were evaluated on one set of 50 positive and 50 negative reviews.

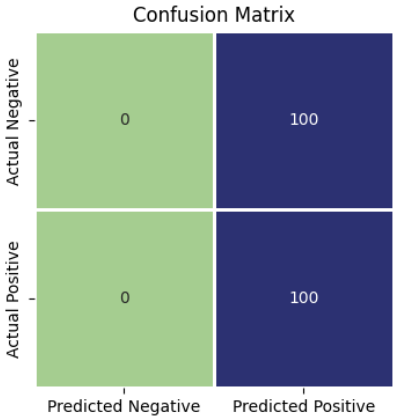
The first confusion matrix on the right belongs to model_v2, and it achieved the following scores:

Accuracy: 0.93
F1 Score: 0.93

The second confusion matrix belongs to model_v1. It is visible that it cannot classify anything as negative. it achieved the following scores:

Accuracy: 0.5
F1 Score: 0.67





5 Discussion

In addressing the difficulty of automated game review production and sentiment classification, the use of GPT-2 variations, specifically GPT2ForSequenceClassification and AutoModelForCausalLM(gpt2), provides different advantages matched to the complexities of the work.

Suitability of GPT-2 for the Task:

The adoption of GPT-2 comes from its fundamental features as a transformer-based language model. GPT-2 succeeds at capturing contextual dependencies inside sequences, making it ideal for applications requiring natural language processing and creation. The autoregressive character of GPT-2, as demonstrated in AutoModelForCausalLM(gpt2), is perfectly aligned with the creative demands of producing coherent and contextually meaningful game reviews.

Simultaneously, the GPT2ForSequenceClassification version, which has been fine-tuned for sentiment analysis, takes advantage of GPT2's ability to grasp intricate relationships within sequences to perform successful sentiment classification.

In addition, transformers, especially GPT-2 in this context, demonstrate excellent suitability for automated game review production and sentiment classification due to their innate capacity to capture subtle contextual connections within data sequences. The transformer architecture's self-attention mechanism enables the model to balance the importance of various sections of the input sequence, making it easier to interpret sophisticated linguistic patterns found in user-generated game reviews.

Transformers' flexibility in handling sequences of varied lengths, together with their inherent parallelizability, make them suitable for processing the complex and frequently dynamic structures of gaming-related textual data. Furthermore, the autoregressive nature of transformer-based language models, such as GPT-2, allows for both sentiment analysis and creative text production tasks, resulting in a cohesive and comprehensive answer to the issues faced by large-scale, user-generated content in gaming. The transformer architecture's performance in a variety of natural language processing applications demonstrates its usefulness, making it a powerful and adaptable tool for jobs that require both quantitative sentiment analysis and qualitative language modeling.

Benefits of the Proposed Model:

- Contextual Understanding: GPT-2's strength relies on its capacity to understand context and capture long-term dependencies within textual material. This is useful in the creation of game reviews, as context is essential for accurately communicating user thoughts and opinions.

- Adaptability through Fine-Tuning: The fine-tuning procedure improves GPT-2's adaptability to the unique peculiarities of game reviews and sentiment analysis. By pre-training on a diverse dataset and then fine-tuning for the job at hand, the model strikes a balance between generality and task-specific relevance.

- Efficiency with LoRA: Including Layer-wise Relevance Adjustment (LoRA) in the fine-tuning process increases efficiency. LoRA's strategy of encoding weight updates with two smaller matrices via low-rank decomposition dramatically reduces the number of trainable parameters, making the model computationally inexpensive while maintaining performance.

- Versatility to handle Sequences: GPT-2's transformer architecture improves at managing sequences of variable durations, which is critical in the context of game evaluations with diverse structures and lengths. This adaptability allows the model to effectively handle user-generated content at varied levels of detail.

- Dual Model design: Using two GPT-2 variations, one for sentiment classification and one for language modeling, demonstrates a dual-task design that covers both the quantitative and qualitative sides of the problem. This thorough methodology ensures

that the model not only correctly classifies feelings, but also creates reviews that truly capture user sentiment.

In contrast to other approaches, the GPT-2-based paradigm provides a cohesive framework that easily integrates sentiment analysis with review production. The suggested model's comprehensive understanding of the context, efficiency benefits through fine-tuning, and adaptability introduced by LoRA all contribute to its viability as an automated game review generating and sentiment classification system. The delicate relationship between these components leads to the model's capacity to manage the challenges of processing massive amounts of user-generated content in the gaming domain.

References

[1] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Brew, J. (2020). Transformers: Cutting-edge Natural Language Processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020.

[2] Liu, B. (2012). Sentiment Analysis and Opinion Mining. Synthesis Lectures on Human Language Technologies, 5(1), 1-167.

[3] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends® in Information Retrieval, 2(1–2), 1-135.

[4] Stone, P. J., Dunphy, D. C., & Smith, M. S. (1966). The General Inquirer: A Computer Approach to Content Analysis. MIT Press.

[5] Huang, Z., Wu, J., Chen, Y., Zhou, M., & Li, Z. (2014). Adapting Sentiment Lexicons with Contextual Valence Shifters. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 635-640.

[6] Lu, B., Ott, M., Cardie, C., & Tsou, B. K. (2011). Multi-aspect Sentiment Analysis with Topic Models. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, 180-190.

[7] Qiu, G., Liu, B., Bu, J., & Chen, C. (2009). Expanding Domain Sentiment Lexicon through Double Propagation. In Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 179-186.

[8] Shaikh, S., Tanveer, M., & Qamar, U. (2008). A Hybrid Approach for Sentiment Analysis in Online Reviews. In Proceedings of the 2008 International Conference on Information and Knowledge Engineering (IKE), 620-625.