**AMAZON VIDEO GAME RATING REVIEWS**

# Motivation and Data Understanding

Among all the reviews, there is still a grey area, the neutral reviews (i.e. a review with a middle rating). People’s perspective and decision are different, and because of this, neutral reviews sometimes tends to be positive and in other cases negative. If a user only gives neutral reviews, deciding which products to suggest to him is difficult and most likely will result in a bad recommendation. The objective of this project is to design a model to perform analysis trained to label the reviews as “negative”, “neutral, and “positive”; after training the model, we will be able to input a neutral review to the model and predict the true sentiment of it.

The data was obtained from [amazon](https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/index.html) [reviews](https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/index.html). The data was scrapped directly from the Amazon webpage, and we will be using the videogames category, selecting any other category would not change the purpose of the project. The original dataset contains 2,565,349 reviews, but we will be using a subsample of 497,577 reviews. The columns of interest are the rating, user ID, and text review.

# Data Preparation

* **Data Loading:** The data was parsed using the code provided in the dataset webpage.
* **Data Cleaning:** Columns of no interest were dropped (e.g. time, upvotes, image, etc.). Then, the punctuations from the reviews and duplicate rows were removed.

Finally, the ratings (1 to 5) were relabeled into 1 and 2 = negative (0), 3 = neutral

(1), and 4 and 5 = positive (2); the final count was “negative” (51984 reviews), “neutral” (46561 reviews), and “positive” (375604 reviews). There is a clear skewness towards the positive reviews.

* **Encoding Words and Labels:** First, create a list with all the words in the reviews to later create a dictionary with the count of each word, this is our vocabulary (352843 unique words). Tokenize (i.e. assign a word a number) the vocabulary for order and lower the computational demand. Then, create a list with each review tokenized and remove any zero-length review. Finally, select a maximum number of words per review (200) to make the padding and truncate the reviews.

# Modeling

* **Data Loaders and Batching:** We select a distribution of 80%, 10%, and 10% for

training, validation, and testing. The initial batch size was 64.

* **Positional Embedding Layer:** First, a “ZeroEmbedding” function was implemented to introduce the biases and reset the weights. Then, to add the positional embeddings to each token (word), we used the class

“PositionalEmbedding”, returning the token with its embeddings and positional embeddings; the embedding dimension was 512, the same as the one pick in paper “Attention is all you need”.

* **Transformer Encoder:** The encoder is defined in the “TransformerEncoder” class, and is composed of one multi-head attention layer (2 attention heads, dimension of intermediate layer of 2014). For the feed-forward, the tokenized text is the input, goes through the position embedding layer, then passes through

the transformer encoder, a dropout layer, and finally through a linear layer with log softmax activation. The activation is log softmax because the prediction is for multi-class and we are using NLLLoss as the loss function.

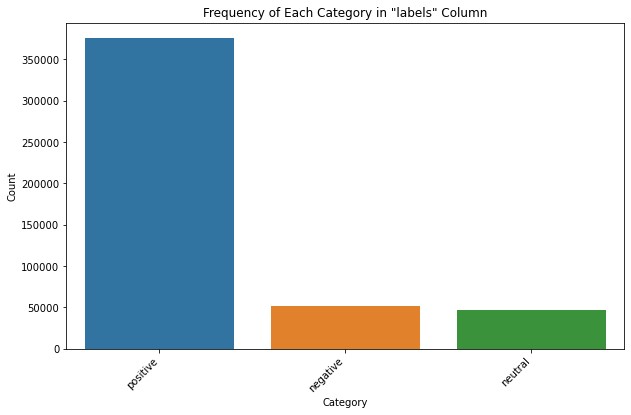
* **Training and Evaluation:** We decided that the benchmark model would be the model described above trained with a learning rate of 0.01 and L2 of 0.001. Later, we played with the hyperparameters (i.e. embedding dimension, learning rate, L2, attention heads, and sequence length).

# Implementation

The first challenge was the disproportionate distribution, to solve this we thought about adjusting it manually but in the end, we decided to leave it as it was, we believe this was a mistake and we will discuss it later. The frequency of each review can be seen in Fig The second challenge was selecting the correct parameters for the base model, but we borrowed some parameters like the embedding dimension from the paper “Attention Is

All You Need”. The final challenge was the computational demand, the encounter of

RAM limitations throughout the entire project, to bypass this we grabbed a subsample of 50K reviews to test everything before running the entire dataset. This solution was very helpful, but the training was still slow and without the GPU was not feasible.



With more time and resources, we would have been able to combine this solution with a recommendation system, as we had the user IDs; and go deeper by applying a decoder to try to predict what a user would write in a review of a recommend game.

# Results and Evaluation

The most important observations, regarding the training, were that incrementing the embedding dimension leads much faster to overfitting taking more computational demand, and that the sequence length and attention heads did not improve nor lead to faster overfitting. The next table presents the results of using different significant hyperparameters (100k reviews) and the metric of evaluation % Test Accuracy:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Epochs** | **Embedding Dim.** | **Attention Heads** | **Learning Rate** | **L2** | **% Test Acc.** |
| 10 | 512 | 2 | 0.01 | 0.001 | 79.4% |
| 10 | 512 | 2 | 0.001 | 0.0001 | 81.1% |
| 8 | 512 | 2 | 0.001 | 0.00001 | 78.7% |
| 8 | 512 | 2 | 0.03 | 0.0001 | 78.3% |
| 6 | 1024 | 4 | 0.001 | 0.0001 | 80.5% |
| 6 | 1024 | 2 | 0.001 | 0.0001 | 79.6% |
| 6 | 512 | 2 | 0.003 | 0.0001 | 82.3% |

These were the highest models among all the others. Finally, we trained a model with the 470k reviews with the hyperparameters from the highest metric, giving a Test

Accuracy of 85.3%. The final score was 5.9 points over our base model, but comparing it with a Bidirectional LSTM Neural Network from [Kaggle](https://www.kaggle.com/code/ademhph/emotion-recognition-using-lstm) with 89% accuracy is 3.7 points lower. While it's true that the Kaggle post classified emotions from tweets, the nature of it was Sentiment Analysis with 6 classes, so there are no excuses for

underperforming.

For now, the results are still shallow, as mentioned above this is the first step of the whole solution. Applying the decoder and merging it with the recommendation system to predict the user review is a cool feature to have. It opens the possibility of offering the users an auto review if they want to which would be based on previous reviews. We hope this solution will increase the number of reviews per customer and with these obtain more feedback to make decisions on the products offered.

# Deployment

The current solution can be deployed to categorize a neutral review (3 stars) as positive or negative, and be able to recommend the user new products. The major issue is the disproportionate distribution mentioned before. The model was trained with this disproportion and that impacts heavily on the final predictions, this can be seen in the confusion matrix in Fig 2, where most of the predictions are labeled as positive, especially neutral reviews. Also, we find some inconsistencies with some words that are present in all types of reviews and could bias the model, later we will see some of them; this could be a product of the different tastes of users or game categories.

**Fig**

**2.**

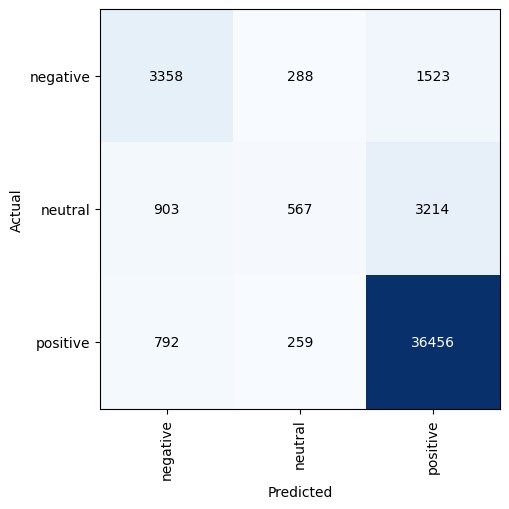
**Confusion**

**Matrix**

**Real**

**vs**

**Predicted**



We do not consider there are any ethical considerations in this first part of the solution, but we think that for the final solution, we should not use the prediction of the conduct of a user for purposes other than the automatic review. If the final solution can produce an accurate custom review with the personality of a user, this can have the potential to be used for other things, like identity thief.

# EXTRA+

While the principal objective was the Sentiment Analysis, we thought that understanding the keywords (nouns) that appeared more in each review label. This would help in knowing if some words are present in more than one review label, and could be

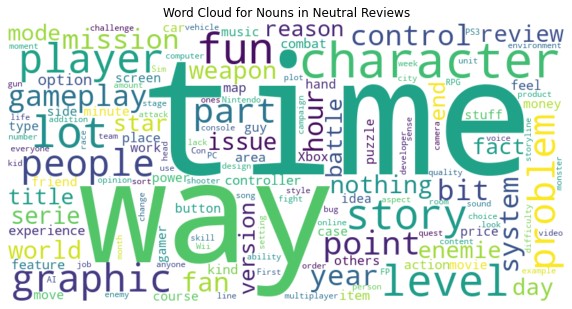
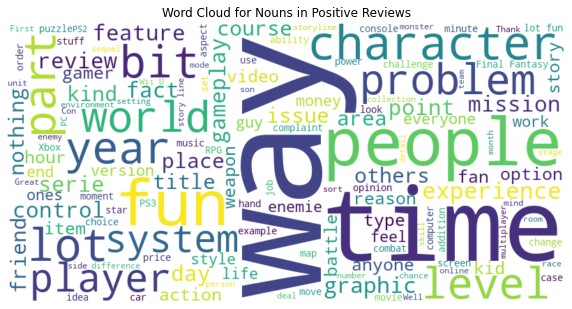
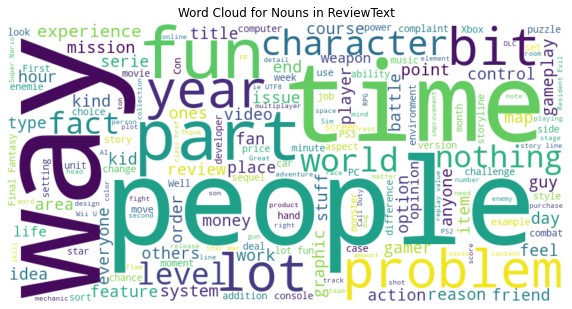
disrupting the predictions.

After Tokenizing the text and identifying nouns for focused analysis and Utilizing NLTK's stopwords to filter out common English words and custom stopwords specific to this

dataset (e.g., 'game', 'play')

# Word Clouds

This technique transforms textual data into graphical representations. We created four word clouds: nouns for overall reviews, nouns over negative reviews, nouns over neutral reviews, and nouns over positive reviews. This can be seen in Fig 3.



**Fig 3. Word Clouds Overall Reviews**.

As mentioned before, some words are present in all types of reviews, and their meanings in each one could differ because of the game category or simply what a person likes. For example, some users may like to spend some “time” with their friends, while others dislike the “time” that takes to upgrade something.

**Sentiment Analysis LSTM:**

Before implementing the Transformer, we developed an LSTM model as a pilot. We used the same parameters as the Transformer Encoder model, but something interesting happened. The initial SentimentRNN model employed specific hyperparameters, including a two-layer LSTM with a dropout of 0.5, a ReLU activation function, and a linear layer for classification. This model achieved an accuracy of 80.3%, utilizing a learning rate of 0.0005, CrossEntropyLoss as the criterion, and the Adam optimizer.

In an attempt to enhance accuracy, a second model was implemented with adjustments such as changing the activation function to ELU, and employing the SGD optimizer with a learning rate of 0.0015. Surprisingly, this modification resulted in a decreased accuracy of 82%.

The unexpected increase in accuracy from 80.3% to 82% when transitioning from the initial SentimentRNN model to the second model with ELU activation and SGD optimizer may be attributed to the non-linear characteristics introduced by ELU, which could better capture nuanced sentiment patterns in the data. Additionally, the change in optimizer and learning rate may have facilitated a more effective exploration of the model parameter space, leading to improved convergence and enhanced predictive performance.

While these considerations are valid, we acknowledge that the observed performance may be influenced by data distribution. We posit that in a more balanced dataset, the Transformer Encoder would likely demonstrate superior performance. Nonetheless, we recommend further exploration, including investigating Bidirectional LSTM and Dynamic RNNs, to comprehensively understand the optimal architecture for the sentiment analysis task at hand.

**Popularity and Product Analysis:**

We selected the Top 1 game (i.e. B00JK00S0S is the ASIN of the game, ID) based on the sum of their rating, their count of positive reviews, and unique reviewers. The percentage of positive reviews was 91.32%. This analysis will be focused on discovering the reason behind its success.

Two advanced techniques were deployed to dissect the text data:

**TF-IDF Vectorization:** Quantify the importance of the words within the positive reviews relative to their frequency.

**Latent Dirichlet Allocation (LDA):** Uncover latent topics by applying an unsupervised learning approach to delineate thematic structures within the reviews.

With the TF-IDF Vectorization, we were able to identify engagement factors, such as the words “worlds”, “level”, and “challenge”. Also, some emotional resonance and game mechanics, such as “emotions”, “feel”, “experience”, “ammo”, “difficulty”, “horror”, and

“stealth”. And distinctive elements, like “Joel” (i.e. a character people liked) or “circumference” (i.e. a feature of something) reflect distinctive attributes that players find memorable.

With LDA Modeling, we were able to find different topics among the reviews, some of them were the narrative and gameplay quality, extrinsic aspects of gaming, emotional depth and immersion, graphics and performances, and social components (e.g.

multiplayer, community).

This analysis provides insights that can directly inform game development to double down on aspects that drive engagement and satisfaction. It also clarifies what customers value most, and the key terms and topics for marketing narratives. The deployment of this tool would be more beneficial for video game developers, but any platform with reviews could offer this as a service for the developers if they do not have the tools or the knowledge to do it.