Developing a Language Model Chatbot for Amazon Video Game Reviews

Alimu Ankaer
Kanav Bengani

alimu.a@northeastern.edu
bengani.k@northeastern.edu

Abstract

This project aims to build an intelligent chatbot capable of answering product-related and review-based questions for Amazon video game reviews. By fine-tuning a large language model on a curated dataset from the Amazon Reviews repository, the chatbot delivers precise and context-aware responses. This project highlights advancements in natural language understanding and demonstrates the practical application of machine learning in consumer decision-making tools.

Overview

This report outlines the development and evaluation of a language model chatbot tailored for video game reviews from Amazon. The chatbot integrates natural language processing techniques with a fine-tuned large language model to provide contextually relevant responses to user queries. The project's significance lies in its ability to help consumers make informed decisions by summarizing key points from extensive product reviews. The approach includes dataset preprocessing, model selection, and iterative fine-tuning to optimize the chatbot's performance. Metrics such as precision and the content of the response are used to evaluate the effectiveness of the chatbot.

Problem Statement

What is the problem?

Navigating through numerous reviews can be overwhelming for consumers. The need for a tool to summarize and answer questions about the product they want, such as their features, quality, or user experience, is important.

Why is this problem interesting?

This problem connects consumer decision-making and artificial intelligence. By addressing it, we create tools that enhance e-commerce interactions, improving efficiency as well as user satisfaction. Use cases extend beyond shopping, such as product research and analyzing market.

What is the approach?

We employ fine-tuning of language models using LlaMa and GPT-Neo on Hugging Face Amazon Reviews Dataset. We attempted to use both models to see how they would differ in performance. The project involves preprocessing the dataset, setting up training pipelines, and optimizing hyperparameters to balance accuracy and computational efficiency. We hoped this approach would work well since the LlaMa model has 1 billion parameters; hence, it should be able to pick up on multiple different patterns efficiently and remember them accurately when the user asks a question.

What is the rationale behind the proposed approach?

Existing methods like search or basic Q&A tools lack the conversational depth and adaptability of large language models. By fine-tuning a model on domain-specific data, our approach achieves more contextual understanding. This allows the user to be able to ask follow-up questions as well.

What are the key components of the approach and results?

In terms of the actual components, our dataset will be loaded into from Hugging Face which has data related to Amazon reviews for video games. As for the model, we had access to a GPT-Neo 125 million parameter as well as Meta LlaMa 3.2-1B parameter model. Our metrics to see whether the approach works will consist mostly of qualitative benchmarking and seeing the relevance of the LLM-generated answer with relation to the question. As for limitations, the model's accuracy may decline for edge cases, such as highly subjective or ambiguous queries. Resource-intensive fine-tuning poses scalability challenges. Also, the definition of precise or related output is hard to measure since there is no clear target accuracy or metric to pursue and quantify.

Experiment Setup

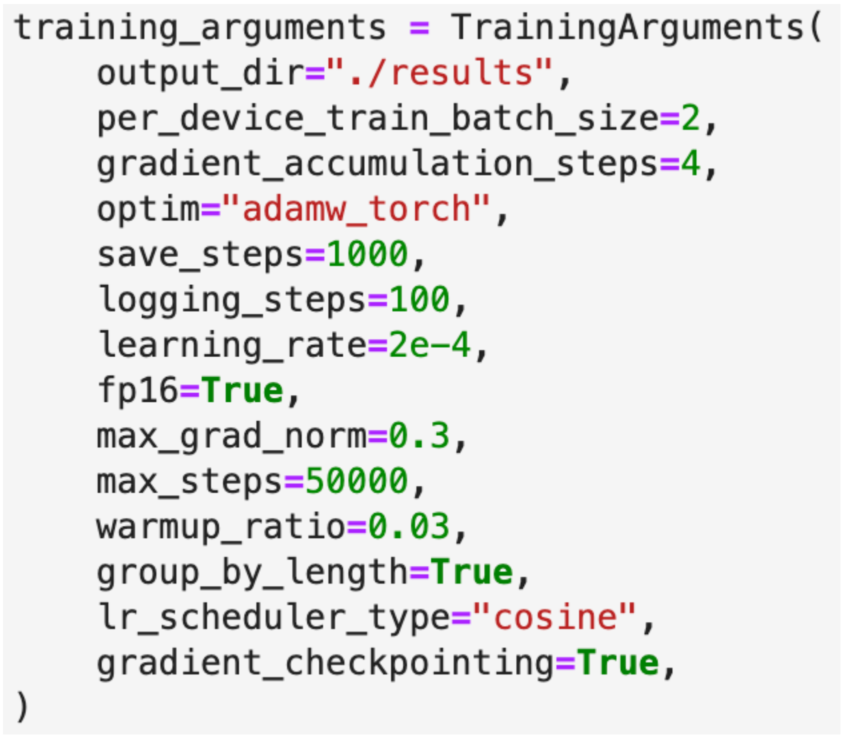
The Hugging Face dataset includes reviews, ratings, and its metadata for video games.

Basic statistics:

- Number of reviews: 4624615
- Number of products: 137269

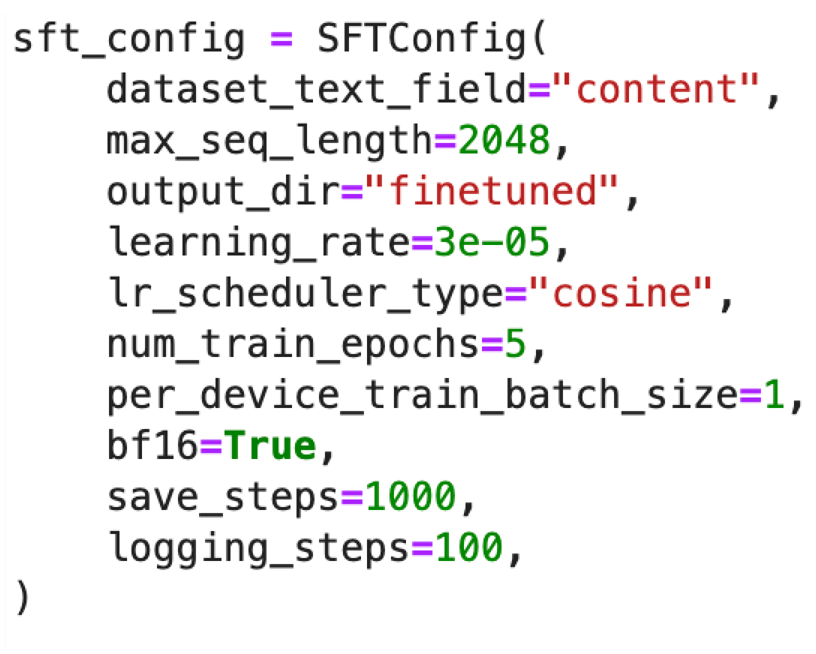
Implementation:

We used a Python-based Hugging Face’s Transformers library for fine-tuning. Key parameters included a learning rate, batch size, lr scheduler type, and number of epochs. Unfortunately, since the dataset was so huge, the number of epochs did not really come into the picture since we had to cut the training short and simply save a checkpoint. We tried running two models. The first being the gpt-neo-125m. This had the following as training parameters:



```python
training_arguments = TrainingArguments(
    output_dir="./results",
    per_device_train_batch_size=2,
    gradient_accumulation_steps=4,
    optim="adamw_torch",
    save_steps=1000,
    logging_steps=100,
    learning_rate=2e-4,
    fp16=True,
    max_grad_norm=0.3,
    max_steps=50000,
    warmup_ratio=0.03,
    group_by_length=True,
    lr_scheduler_type="cosine",
    gradient_checkpointing=True,
)
```

The second model that we ended up going with was Meta's LlaMa 3.2-1B. The training parameters were the following:



```
sft_config = SFTConfig(
    dataset_text_field="content",
    max_seq_length=2048,
    output_dir="finetuned",
    learning_rate=3e-05,
    lr_scheduler_type="cosine",
    num_train_epochs=5,
    per_device_train_batch_size=1,
    bf16=True,
    save_steps=1000,
    logging_steps=100,
)
```

In terms of the computing environment, we trained the models on a cluster that was configured with 1 GPU, 8 CPUs, 48G RAM.

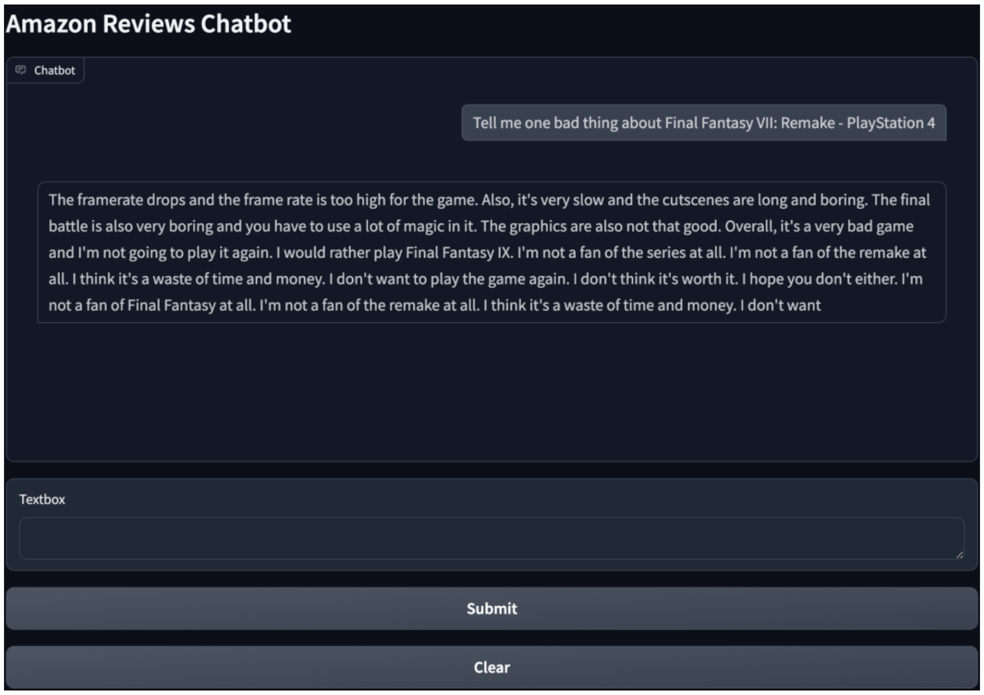
End-to-End Flow

The original data consists of two parts, one is Reviews, and the other is Metadata. we use [text] column from the Reviews and [title] column from the Metadata, then joined them based on the only related column parent_asin (parent ID of the product), so the final data format before we start tokenization is f"Product: {product title} \nReview: {review text}". Once tokenized, each of these product-review pairs gets fed into the model and loss is attempted to be minimized with each step through gradient updates. With both models, we had to cut training time short unfortunately, so not all of the data was seen by the model.

Experiment Results

Main Results

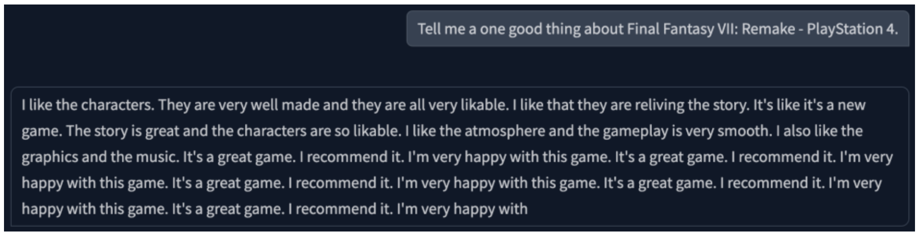
The result is shown below. In this example, a question is asked about a certain game, and it comes back with an accurate response. Some things to note in this example, however, is that it does not have an end token, which leads to hallucination towards the end. Therefore, we instated a max token length of 200 for this case.


Amazon Reviews Chatbot

Tell me one bad thing about Final Fantasy VII: Remake - PlayStation 4

The framerate drops and the frame rate is too high for the game. Also, it's very slow and the cutscenes are long and boring. The final battle is also very boring and you have to use a lot of magic in it. The graphics are also not that good. Overall, it's a very bad game and I'm not going to play it again. I would rather play Final Fantasy IX. I'm not a fan of the series at all. I'm not a fan of the remake at all. I think it's a waste of time and money. I don't want to play the game again. I don't think it's worth it. I hope you don't either. I'm not a fan of Final Fantasy at all. I'm not a fan of the remake at all. I think it's a waste of time and money. I don't want

Below is an example of what happens when the chatbot ends up hallucinating. This could have happened for various reasons: there was not enough information in the training dataset about this specific question, the model was not trained for long enough, et cetera.


Tell me a one good thing about Final Fantasy VII: Remake - PlayStation 4.

I like the characters. They are very well made and they are all very likable. I like that they are reliving the story. It's like it's a new game. The story is great and the characters are so likable. I like the atmosphere and the gameplay is very smooth. I also like the graphics and the music. It's a great game. I recommend it. I'm very happy with this game. It's a great game. I recommend it. I'm very happy with this game. It's a great game. I recommend it. I'm very happy with this game. It's a great game. I recommend it. I'm very happy with this game. It's a great game. I recommend it. I'm very happy with

Supplementary Results

As for the parameter choices, we kept the learning rate stagnant at 3e-5 for all the runs since it was relatively low (good for a huge dataset) and it was used in existing work as well. The batch size was something we played around with. Unfortunately, higher batch sizes did start leading to memory issues, which is why we kept the batch size to 1 to avoid any such issues. The learning rate schedule type was set to cosine as this usually helps in capturing semantic meaning. As mentioned above, though the number of epochs was set to 5, it did not end up being that useful since our code did not go through till then.

Discussion

With certain examples as shown in the first picture, the chatbot ended up doing pretty well. Unfortunately, there were a lot of cases where it was not outputting what we would deem qualitatively “good” responses. There are a couple of things we could do to improve this. First, we could try to use parameter-efficient fine-tuning. This not only will help reduce training times, but it couuld also help retain the some important weights that the LlaMa 3.2-1B model already had pretty accurate.

Additionally, we can also possibly try doing a little bit of retrieval augmented generation by extracting context from the database related to the user-asked question and giving that as context to the LLM before it comes up with an answer. This will drastically improve the relevance of the answers as there is a higher chance the LLM remains on topic. Feeding a good system prompt may also help in ensuring the LLM remains on topic; for instance, it will better be able to refuse to answer when a questions is asked by the user that is unrelated to what the LLM has been trained on. Lastly, improving benchmarking techniques is a great way to open this project for extensions in the future. If we are able to create some benchmarks, this problem can be continued to be improved upon by basing the performance of the new iterations based on these.

Conclusion

In this project, we developed and evaluated an intelligent chatbot designed to answer product-related and review-based questions for Amazon video game reviews. By fine-tuning large language models like GPT-Neo and Meta's LlaMa on a curated dataset, we demonstrated the potential of large language models in enhancing consumer decision-making. Our findings show the utility of NLP techniques in summarizing vast amounts of content and providing contextually relevant responses.

Despite the promising results, limitations such as hallucination, incomplete training due to resource constraints, and challenges in qualitative evaluation highlight the need for further optimization. Future work could explore parameter-efficient fine-tuning, retrieval augmented generation, and improved system prompts to address these issues. Additionally, establishing robust benchmarks would enable systematic evaluation and refinement of the chatbot over time.

Ultimately, this project represents a meaningful step toward integrating advanced tools into e-commerce, showcasing the power of artificial intelligence, specifically generative models to transform how consumers interact with and derive value from product reviews.

References

- Hugging Face Amazon Reviews Dataset: link
- Hugging Face Transformers Library: Documentation
- PyTorch: Website