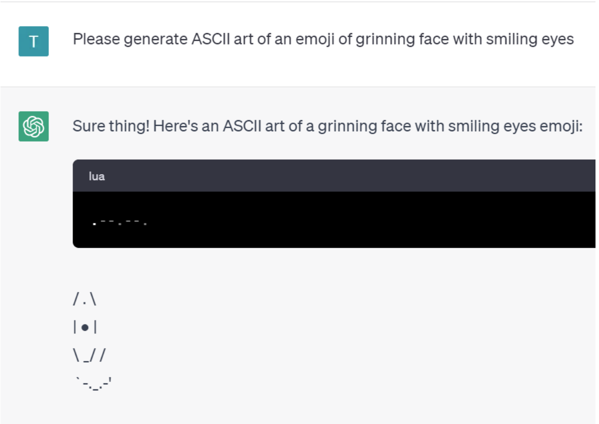
Redefining Online Communication: Unique Artificially Created Emojis

Daniel Cufiño, Tochukwu Okwuonu
Department of Computer Science, Rice University
tmo6@rice.edu, dc58@rice.edu

Abstract

The use of emojis in online communication has increased since the advent of smartphones and social media. Currently, the emojis available for use for users are a predefined set that is designed by the platform they are using. Apple, Android, and Facebook each have their own. This allows us to have a large dataset of prelabeled emojis. These labels can be considered very accurate because they are the labels provided by the companies that made them. Our focus for this project is to create a model that can create ASCII-based emojis derived from a text description. In our project, we explore the use of a decoder transformer where the input text, is tokenized and fed into the model one at a time and the model uses masked self-attention to train on predicting the next character. We will train our model on ASCII emoji versions provided by Apple, Android, and Facebook, and judge its performance based on its similarity to standard emojis.



T Please generate ASCII art of an emoji of grinning face with smiling eyes

Sure thing! Here's an ASCII art of a grinning face with smiling eyes emoji:

```
lua
.--.--.
```

```
/.\
|•|
\_//
`-._.-'
```

Figure 1. Here we show our attempts to use one of the best language models available ChatGPT, which uses, GPT 3.5. Our objective is to generate responses that are better than the ones created by ChatGPT. These results can be generated by the public using ChatGPT.

1. Introduction

Creating ASCII images based on a text description is a challenging task for computers. It requires identifying the key emotions and wording from the text description and then using characters to create art. This is especially difficult because of the high room for error in text-based images. Based on our research there are no models that specifically convert text description to ASCII art. Most of them focus on converting images into ASCII art. As shown in Figure 1, using other language models such as ChatGPT, are wildly inaccurate in creating ASCII images. They show the potential to be useful in creating ASCII emojis. Our work will be focused on fine-tuning GPT-2 and training it to create accurate ASCII emojis.

2. Related Work

The idea of next-word prediction has been around for a while in places from the next-word prediction feature of a keyboard app or the search autofill suggestion in Google Search. GPT-2 is a large transformer decoder model with over 1.5 billion parameters pre-trained on a large dataset of 8 million web pages. The pretraining process uses unsupervised learning techniques to predict missing words or tokens in a given sentence or passage. It was trained to learn different tasks such as question answering, reading comprehension, summarizing, and translation from raw text, using no task-specific training data [1]. This development in the language models has been general for chatting or answering questions, and no focus has been put into generating ASCII images. The only other similar work to our project is the text-to-image generation models such as Imagen. Imagen is a text-to-image diffusion model with a very high degree of photorealism and a deep level of language understanding. Imagen builds on the power of large transformer language models in understanding text and hinges on the strength of diffusion models in high-fidelity image generation. This allows it to generate an image from text[2]. The work done separately for text-to-text models and text-to-image models separately served as inspirations for our work[3]. However, these models are so general and neither of them translates well when asked to generate ASCII images.

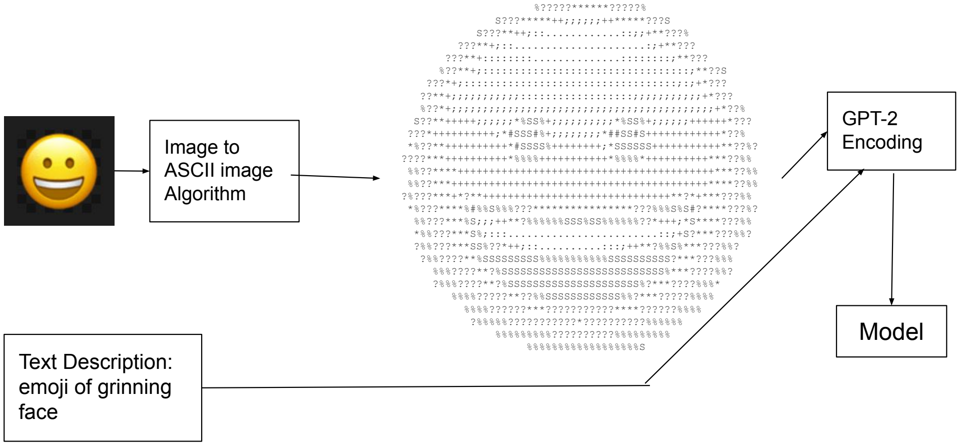


Image to ASCII image Algorithm
GPT-2 Encoding
Model
Text Description: emoji of grinning face

Figure 2. Above depicts the result of converting a sample image in our dataset to its corresponding ASCII art through the use of our algorithm.

3. Proposal

We propose fine-tuning a pre-trained GPT-2 model and training it to accurately return ASCII versions of emojis. Our goal is to build on it as a foundation, and then train it on the specific task of creating ASCII images. We believe that this will be enough to accurately create the results we've hypothesized.

4. Model and Data Preparation

We use a publicly available dataset found online through Kaggle that contains a CSV file of every unique emoji created by multiple companies such as Meta, Google, and Twitter as well as the associated description of the image.[1]

We used PyTorch to implement our training and used the hugging-face transformers library to access a publicly available implementation of the GPT-2 mode.[2]. Specifically, we used the pre-trained weights of the GPT-2 medium model which has 335 million parameters. Before converting the images to ASCII, they had to be resized to be 50x50 to ensure the number of tokens for input did not exceed 1024, which is the maximum sequence length the model can be used with. We then removed all emojis whose structure wasn't conventional for use on social media, as well as GIF emojis. We decided to remove these as we are primarily interested in generating ASCII art of emojis that would be shared on social media platforms and the non-conforming design of other types of emojis may adversely impact our training.

Once the data was pre-processed, we wrote an algorithm that converts images to ASCII art by mapping each pixel to a commonly used ASCII art symbol based on the brightness of the pixel. With the images now being represented in a format compatible with our text-based model, we are able to prepare a prompt for the model to train on by creating a string asking the model to generate an ASCII art emoji describing the image in question concatenated with the corresponding ASCII art. This string is then tokenized using the GPT-2 tokenizer, resulting in a sequence that embeds the art represented as token IDs. When fine-tuning, we do not mask any of the input tokens because we want the model to be able to generate text based on all provided inputs such that it learns how to generate the next sequence of text representing the provided prompt.

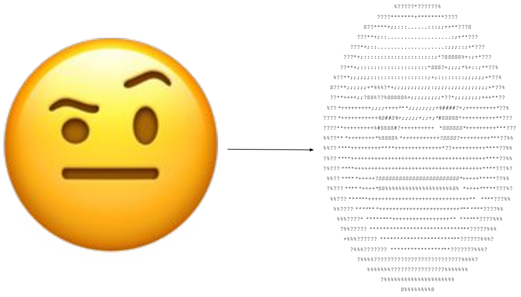


Figure 3. Above depicts the result of converting a sample image in our dataset to its corresponding ASCII art through the use of our algorithm.

5. Experimenting and Results

As mentioned in previous sections, the results generated by currently existing large language models like ChatGPT produce unsatisfactory results. Due to this, we wanted to perform smaller-scale training at first to see if GPT mod-

[1] Dataset

[2] GPT-2 Documentation

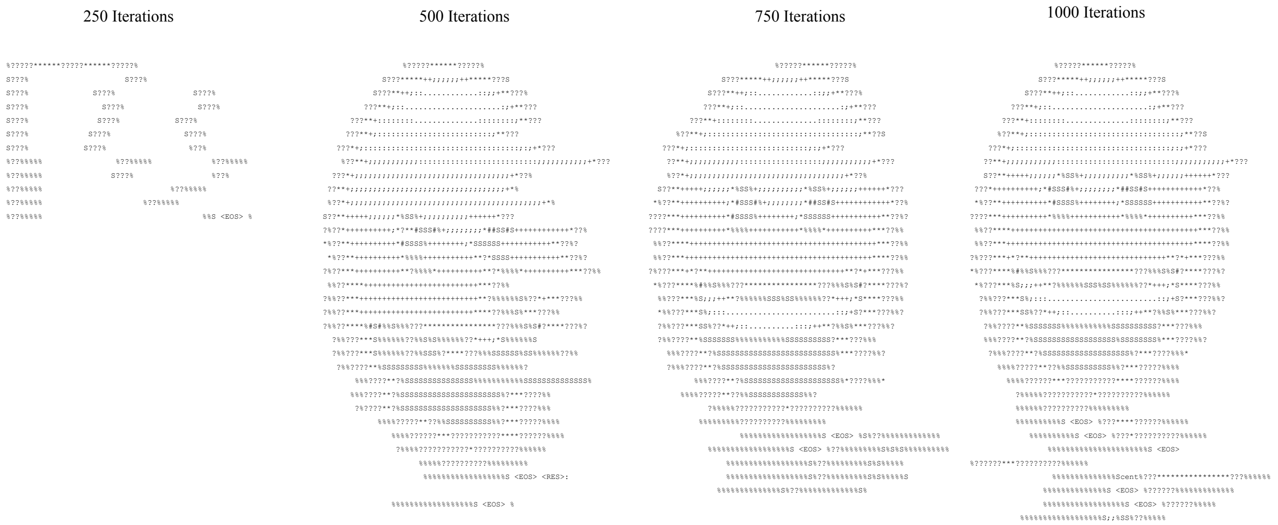


Figure 4. Result of training our model on a single sample from our dataset of numerous iterations

els are even capable of properly learning the task at hand. This resulted in training on a single sample in our data set, keeping track of the number of iterations until an acceptable result was produced by our model. All training done was using the Adam optimizer with an initial learning rate set to 3^{-5}. We also only trained using mini-batches of size two as the compute available to us at the time didn't have enough memory for batch sizes of any greater size.

After numerous trials on different images, we found 1000 training iterations to be the minimum number of iterations required to produce results that are reasonably representative of the true original image. However, the results produced by our model aren't entirely accurate and they tend to have some random noise near the bottom of the image as shown in Figure Five. With that said, we were able to determine that this model is at least capable of learning the task at hand.

After determining that GPT-2 was capable of being fine-tuned to at least memorize this task, we now sought to determine whether it was able to generalize the structure of the art it was being trained on. We do this by creating a hand-picked subset of data and training our model on it. If the model can generalize, then we expect it to be able to produce unique art. This can be tested by passing a prompt to our model that encompasses features from two or more different emojis that were trained on.

We train on six images, five of which are variations of an emoji with a large grinning smile and the sixth one of an image of a raised eyebrow emoji. We train for 1000 epochs, or until we reach an average epoch loss of less than 0.01. This training was done on an NVIDIA A100 Ampere GPU and took 25 minutes to train on just these six images. The results from this subset testing were quite promising. As evidenced in Figure five, our model was able to generalize features from different and combine them to create ASCII art of an entirely unique emoji. This indicates that the GPT-2 model is capable of being fine-tuned at a larger scale on our entire dataset to be able to generate even more robust emoji ASCII art.

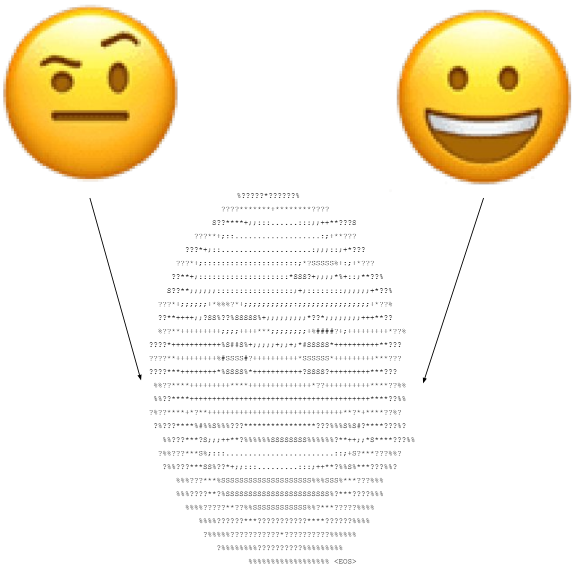


Figure 5. Above depicts our fine-tuned model generating ASCII art for a new and unique emoji by combining features of other emojis it was trained on. The prompt used to generate this output was, "Please generate ASCII art of an emoji of grinning with a raised eyebrow." It should be noted that there were trailing padding tokens that were generated but removed for visual purposes.

At this point, we sought to generate some empirical/statistical evidence of the efficacy of our fine-tuning.

However, because the output of our model is inherently generative, we're not able to use conventional metrics like accuracy or F1. We decided to use a survey method to create a performance benchmark for our model[3]. In order to ensure the validity of our survey, we walked around campus and asked every fifth random person the question, ”Based on the prompt, ”Please generate ASCII art of an emoji of grinning face with a raised eyebrow”, rate the quality of the result on a scale of one to ten”. We then showed them the art generated in Figure 5. We ensured that the person asking the question did not know the person answering the question in order to limit bias from the person providing the answer. After 40 students we felt that we had a large enough sample size in order to assess our model. As shown in Figure 6, the feedback we received from students skewed more positively, with over 80% of the feedback provided rating the performance over an 8 out of 10, with the lowest rating given being a 5. Based on this evidence, we believe that it indicates that our model performed well and up to the expectations we held before beginning the experiment.

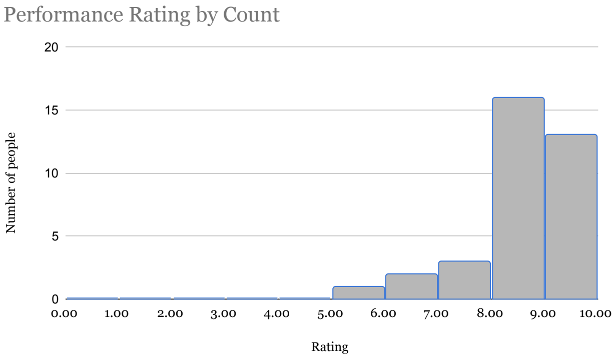


Figure 6. The above depicts the results of our polling of random students. The x-axis represents the rating that they gave. The y-axis represents the count of students that gave each rating.

6. Future Work

The biggest limitation we faced while working on the project was the GPU capacity of Google Colab Pro while training the GPT-2 model. Based on our observation, Colab Pro provides two instances of GPU, one with 16 GB of GPU RAM and another with 40 GB of GPU RAM. The instance with the 16 GB of GPU space could only handle training the model on 1 image at a time, while the instance with 40 GB of GPU space could only handle training 4 images at a time. The limited GPU RAM made it impossible for us fully train a single model instance on all the images in order to test the breadth of its processing. We would have been able to further test, its ability to combine more emojis from the prompt. However, we believe that despite the limited GPU RAM, we have proved the ability of our model to create unique ASCII images from text-based on the provided training data. This is shown in Figure 5, where we were able to successfully combine the grinning face emoji and the raised eyebrow emoji into one ASCII emoji with the prompt, ”Please generate ASCII art of an emoji of grinning with a raised eyebrow”.

In the future, based on the current model that we have provided if we gain access to more GPU RAM, we can be able to train the model to have more emojis. We also intend on adding the metric of where the emoji comes from such as Apple, Facebook, or GMail, and making sure our model can differentiate the different ASCII emojis. We also think this model can be used to create an app or website that uses its abilities, in order to gain more performance feedback to do further fine-tuning.

References

[1] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.

[2] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *ArXiv*, abs/2205.11487, 2022.

[3] Y. Takeuchi, D. Takafuji, Y. Ito, and K. Nakano. Ascii art generation using the local exhaustive search on the gpu. *2013 First International Symposium on Computing and Networking*, pages 194–200, 2013.

[3] Survey Methodology