Emoji Image Generation Using GANs: A Detailed Analysis

I. Introduction

In the digital age, communication has transcended traditional linguistic boundaries, giving rise to new expressive tools. Emojis are small digital images or icons, used in electronic communication to express an idea, emotion, or concept. From the smiley face to a vast array of objects, symbols, and expressive faces, emojis have become an integral part of our digital conversations, bridging language gaps and adding emotional nuances to text-based communication. In effect, they are a universal language of the digital era, transcending cultural and linguistic borders. Given the rising popularity and importance of emojis in modern communication, creating new and diverse emojis that cater to a wide range of emotions, ideas, and cultures becomes crucial. However, generating these images manually is time-consuming, requires artistic skills, and may still lack the diversity and inclusivity needed in a global communication context. To overcome these challenges, we decided to use our knowledge of Machine learning. More specifically, we used a class of AI algorithms known as Generative Adversarial Networks to automate and diversify emoji creation. GANs are known for their ability to create new, synthetic data that resembles the input data, making them an ideal choice for our task of emoji generation. For the task at hand, we used a dataset sourced from Kaggle titled 'Emoji Image Dataset'. It contains over 20,000 emoji images from a variety of platforms including Apple, Google, Twitter, and Facebook. These platforms offer diverse artistic styles and interpretations of emojis, presenting a rich data source for our project. By automating emoji creation with machine learning, we can not only expedite the process but also introduce an unprecedented level of diversity in design and representation.

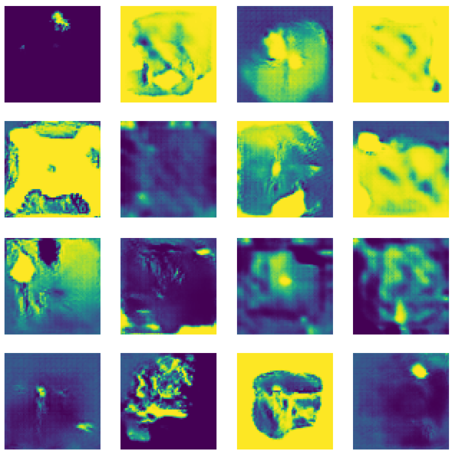
II. Data Analysis and Preprocessing

The 'Emoji Image Dataset' from Kaggle was an ideal choice for our study. It contains a wide array of emojis derived from multiple platforms such as Apple, Google, Twitter, and Facebook. This wide variety offers a collection of different artistic styles and interpretations of similar emoji themes, which provides us with a diverse base of images to learn from and replicate in our generated emojis. These images varied in sizes, color palettes, and artistic styles. Some were simplistic in design, while others were quite detailed, reflecting the different design philosophies of the platforms they were sourced from. Our first task was to perform a comprehensive analysis of this data to understand its structure, diversity, and potential challenges in using it for our GAN model. Through this analysis, we found the need to implement specific preprocessing steps to make the data suitable for our GAN architecture. To handle the variation in image sizes, we resized all images to a fixed size of 72x72 pixels. This uniformity is essential for feeding the data into our GAN model, which requires consistent input dimensions. Given the large size of the dataset, loading all images at once could potentially lead to memory management issues. To prevent this, we utilized a batch loading strategy where only a subset of images was loaded at a time. This strategy was an essential choice considering computational limitations and efficiency. The color values of the images, originally in the range of 0-255 (standard for RGB images), were normalized to a range of -1 to 1. This step is vital for GANs, as normalized inputs can help the model converge faster during training and also improve the overall stability of the model. By conducting a thorough analysis of the dataset and understanding its intricacies, we were able to make informed decisions about the preprocessing steps. These decisions played a crucial role in building a more effective and efficient GAN model. Understanding the data is a pivotal part of machine learning and AI, and our analysis reflects the same. The decisions derived from our analysis not only helped us handle the data but also informed our choices in the subsequent model building stage.

III. Methodology: Generative Adversarial Networks (GANs)

Generative Adversarial Networks, or GANs, are a class of artificial intelligence algorithms used in unsupervised machine learning. GANs are known for their ability to generate new, synthetic data that resembles the input data, making them a perfect choice for our task of emoji generation.
GANs consist of two distinct models, a Generator and a Discriminator, which are trained together. The Generator creates new data instances, while the Discriminator evaluates them for authenticity. It decides whether each instance of data it reviews belongs to the actual training dataset or not. The first component of our GAN is the Generator model. This model takes a random noise signal as input and generates an image as output. The goal of the Generator is to produce images that are indistinguishable from the original dataset in the eyes of the Discriminator. We constructed our Generator using several layers. The Dense Layer is a standard layer type that works well for many types of problems. It is particularly suitable here to take our input and distribute it across the network. The Batch Normalization Layer normalizes the activations of the previous layer at each batch, it applies a transformation that maintains the mean activation close to 0 and the activation standard deviation close to 1. It improves the speed, performance, and stability of the GAN. The Leaky ReLu Activation Function is a variant of the standard ReLu function that prevents the "dying ReLu" problem by allowing small negative values when the input is less than zero. It enables the network to learn complex patterns efficiently. The Convolutional2D layers are typically used in image processing problems to learn spatial hierarchies of features. Here, they help the Generator understand and reproduce the spatial structures within the emoji images.

The second component of our GAN is the Discriminator. This model takes an image as input and outputs the probability of the image being a real image from the dataset. The goal of the Discriminator is to correctly classify the real and generated images. We constructed our Discriminator also using several layers. The Convolutional 2D Layers help the Discriminator understand and assess the spatial structures within the images. They form the basis of the Discriminator's ability to evaluate the authenticity of the images. The Leaky ReLu Activation Functions is used for the same reasons as in the Generator, these functions enable the Discriminator to learn efficiently. The Dropout layer is a regularization technique that prevents overfitting by randomly setting a fraction of input units to 0 during training. It helps the Discriminator generalize well and not overfit to the training data. The Flatten Layer flattens the input and prepares the vectorized data to be input into the Dense layer. The Dense Layer classifies the flattened features into real or fake categories. By carefully selecting and combining these layers, we built a robust GAN capable of learning from the diverse range of emoji styles in our dataset and generating new, believable emojis. The choices made in building the model were driven by the nature of our data and the specific requirements of the GAN architecture.



IV. Results

During the training stage, we ran the GAN for 50 epochs, taking approximately 2.5 hours. During each epoch of training, we generated and saved each resulting image. The generator starts by generating very low quality, blurry images that are easy to discern for the discriminator. The image generation function used the model to generate a batch of images, normalized them and saved for review. This function allowed us to track the progress of the GAN as it went through each epoch. We were able to see how the generator progressively improved at generating realistic emojis. After examining the GAN’s process of generating images, we noticed a few things. First, the images produced greatly improved in clarity compared to the first epoch. The first images produced by the GAN had no shape, direction and could not really be distinguished from the random noise that we fed into the generator. Next, the generation time was steadily decreasing over the time of training. Lastly, one thing that was included in the code that helped us with adjusting parameters was that the model saved every 15 epochs. If changes needed to be made, we would not have to retrain the entire mode. We were left with images that somewhat resembled the emojis in our original dataset. Given our computational power and time for the project, I think our output can be considered a success.

V. Reflection

Reflecting on this project, it definitely greatly broadened our understanding on GANs and image generation models. There is so much potential in this field, and it is still in its early stages. On the other hand, as straightforward as generating one set of images sound, it came with a lot of underlying problems that were new to use, and needed a solution to get around it. Data preprocessing played a crucial role in our project. Understanding the data and vital elements that lie within really helped structure our preprocessing step and made it possible to feed these images into the model. Building the GAN itself deepened our understanding of neural network design and optimization. We learned about the merge between the generator and discriminator's tasks, and how it eventually leads to realistic images generated by the model. Choosing the right layers, sizes, and activation functions contributed to the somewhat success of our GAN model. Observing the model generate images through multiple different epochs gave us huge insight into how GANs generate images and what the process actually looks like. Finally, we were introduced to the real potential of AI, in image generation. We have seen AI being widely used in other sorts of generative ways, text/sequence generation for example, and this was a good introduction to how we can use it ourselves. In conclusion, we think we succeeded in building a model to successfully generate emojis. It provided us with practical experience in building GANs, and gave us a solid result to share with others.