CS230

Pokémon Generator

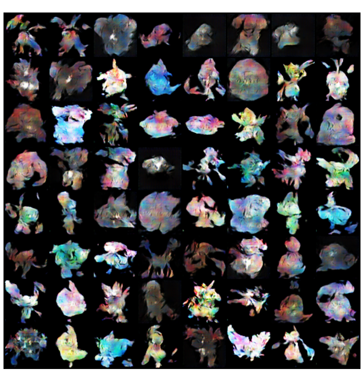
Rebecca Wong
rwong01@stanford.edu

1 Introduction

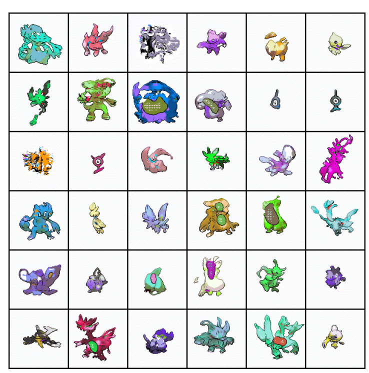
The Pokémon franchise is worth an estimated $105B USD, the largest media franchise in the world, but to date only contains roughly 800 characters. In this project, we aim to use deep learning to create new Pokémon character sprites as well as names for each character. We will use StyleGAN variants from NVIDIA to do the sprite image generation. The input to our algorithm is a collection of existing Pokémon sprites (images) and their names, which is a limited dataset of less than 1000 images. We then use a neural network to output a new Pokémon and new names. The ultimate goal is to generate realistic-looking Pokémon that could be mistaken for real Pokémon by a casual player. Beyond the entertainment value of this project, successfully being able to generate new characters could open many media franchises (Hello Kitty, Mickey Mouse, Anime, etc) to new depths of content creation at a fraction of the time and cost currently required.

2 Related Work

Several previous works have attempted to achieve a similar goal of generating new Pokémon, with varying degrees of success. Several works using DCGAN and WGAN seem to output relatively poor performance, while AEGAN and StyleGAN seem to produce at more decent-looking results. Style-transfer based approaches seemed to be the best starting point as we recognize the style of our favorite characters very easily and this can be a clear giveaway of generated images even if the character is otherwise well-formed. Previous works do not seem to be able produce high resolution sprites that are detailed enough to be mistaken for real Pokémon so this was the primary objective in this work.



(a) Results from previous DCGAN work



(b) Results from previous AEGAN work

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

3 Dataset and Features

There are approximately 800-900 official Pokémon characters in the Pokédex which are accessible via Kaggle and Bulbapedia. For a standard GAN, this is a very small data set. However, the recent advances in StyleGan2 with adaptive discriminator augmentation (ADA) indicate that this can help reduce overfitting in GANs when the data set is small. We use standard horizontal mirroring for data augmentation as well as testing various other augmentation techniques available through StyleGAN2 such as pixel blittng, geometric transformations, and color shifts. The data set comes from a Pokémon database where I used an image scraper to download 898 Pokémon images. The images were not all uniform in size so I used an image resizer to force ever image to a square 128x128px image. Furthermore, the images were PNGs with 4 channels RGBA (transparency layer) while the model needed 3 channel RGB so I additionally flattened all images and remapped into 3 color channels. In my original dataset, I noticed there were discrepancies between the style of the sprite (8-bit style vs high-resolution) and so I had to recreate my dataset after several days of training and transfer the model over. I also re-flattened the dataset onto a white background which made the results look much better than originally on black backgrounds. For the text dataset, I used a list of the 898 Pokémon names, including special symbols. While in my original proposal I intended to use description text alongside the Pokémon, this turned out to not be a standardized data product that I could pull from the Bulbapedia API so I decided to focus more on the image generation and name generation.

4 Methods

I started with the image generation portion of the project since this seemed to be the most difficult part based on the results founds previously that were subpar. I used the StyleGAN2-ADA library and fed it my Pokémon images and tested a couple different augmentation parameters as well as tuning the gamma, batch size, and optimizer functions. I ran the model on my personal machine which contains 32GB RAM and a GeForce 3060Ti graphics card with 8GB of VRAM. Each tick ran for about 3 minutes, and output a sample snapshot every 10 ticks. The training speed was bounding by my single GPU and 8GB of VRAM. In total, I eneded up training for about a week to reach my final results, but adjusted paramters at different stages to tune the results. On the text generation, I was able to adapt our earlier homework assignment for dinosuar names into Pokémon names and ran the model for 30-75,000 iterations with a sweet spot around 50-60,000.

5 Tuning

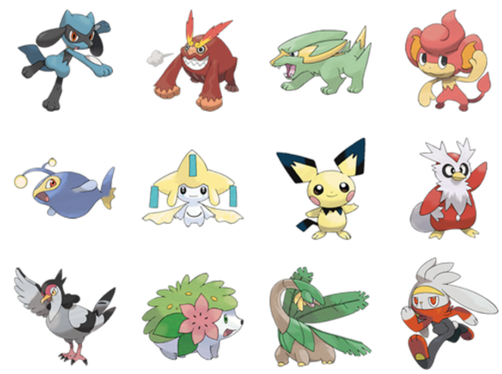
Key parameters that I tuned were the choice of R1 regularization gamma, augmentation techniques at different stages of training, and batch sizes. I tested gamma choices between 0.1 and 100, and found best results around 15-25. When gamma values were too high, the model was much less stable and introduced many more artifacts, while then the gamma was too low, the shapes did not adapt fast enough. The other key choices were with regards to what kinds of augmentations to use and at what stage in the generation. I found that using all of the augmentation filters could lead to too many artifacts showing up, while not using enough would cause the results to look too similar to the input data set. Additionally, I used early stopping to prevent the model from becoming too uniform which resulted in very monochrome samples as seen in Figure (c), and then I would restart the training again.

6 Results

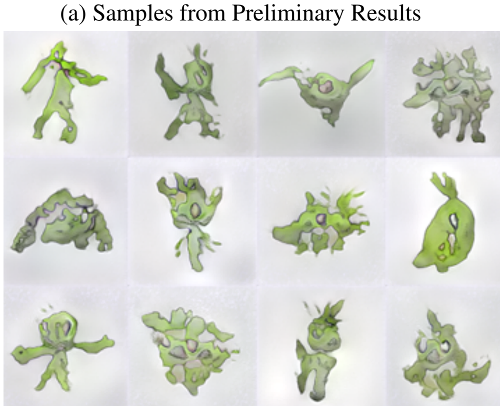
Samples from my final results can be seen below in Figure (d). Overall, I believe the model worked quite successfully and that the generated Pokémon are quite convincing. The shapes and coloring were the most successful, while the eyes and appendages had more varied success. Early training results as seen in Figure (a) were able to replicate color schemes from Pokémon, but lacked detailing to make a convincing character, whereas later stage training was able to create more cohesive details like wings, legs, tails, and eyes. One of the challenges with the results is the subjectivity of what looks "good" since there was no numerical metric to look at to decide when to stop training or which results were good enough. For some characters, early stopping was more useful to prevent overfitting, while for others, even after a lot of training still had consider improvements to make.

6.1 Sample Generated Images

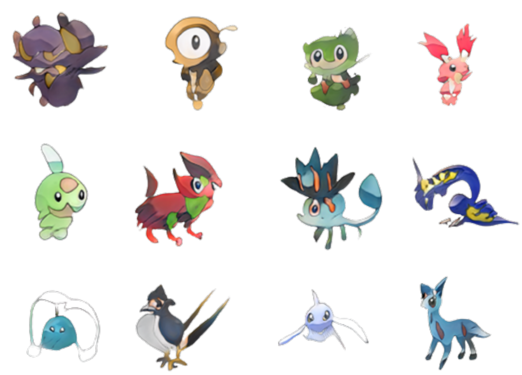
(a) Samples from Preliminary Results



(b) Samples from Original Dataset



(c) Monochrome effect from lack of augmentations



(d) Sample from Final Results

6.2 Sample Generated Names

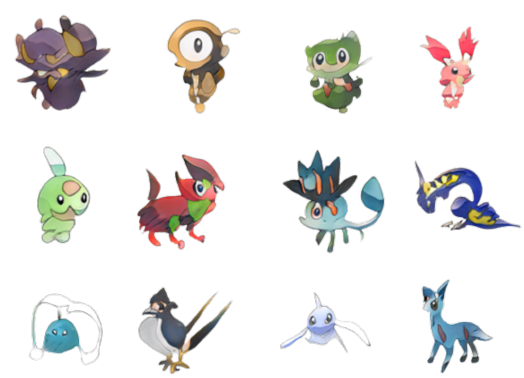
- Mewtro
- Gonaa
- Gutzok
- Macalos
- Zutuff
- Cabgot
- Wrublion
- Baltri
- Vomles
- Worbole
- Baduth
- Vickish
- Altte
- Ramcoo

6.3 Sample Real Names

- Buneary
- Forretress
- Yamask
- Wailmer
- Sinistea
- Tornadus
- Malamar
- Hatenna
- Piloswine
- Ekans
- Pachirisu
- Kyogre

7 Evaluation

Since there is no ultimate objective function that I am evaluating against, it was more important to me to see how the GAN was progressing in terms of a few key features: shape definition, appendages and textures, facial features and details, and color. While some samples were quite good as seen above in Figure (d), others still had a high degree of artifacts showing up which make the characters look more like aliens than animal-based. With these, the shapes were still usually quite good, but the facial features were lacking and made it look very strange. In some cases, changing the augmentation helped to reduce the amount of noise that was being fed in to lead to images with fewer artifacts.



(a) Good samples from Final Results

(b) Poor samples from Final Results

I was able to fool some people with my generated Pokémon by either thinking mine were real or that real ones were generated, so while the results are not perfect, I believe this is a successful metric that previous works were not able to achieve. The name generation was also very believable and could even be made into a game of guessing if the name is real or fake.

8 Future Work

There are several avenues that could be pursued as part of further work expanding on these results, specifically around the facial features and generating more contextual data for the characters.

1. It would be interesting to deep dive into the layers and freeze layers that are responsible for overall shapes and colors as these seem to be quite good across a large majority of the samples. Then we could just train on the details of the eyes, ears, mouths, and minor shading details which would greatly improve the structure of the characters.
2. Integrating more character data into text generation. As mentioned, this was part of my original proposal, but was ultimately unrealizable due to the lack of standardized training data. I believe it would still be a very nove path to go down, but would require substantially more time developing the dataset. A modified version of this could be to just work with the Pokémon types and moves and then generating text around those individual characteristics.

9 Conclusion

In this project, we successfully demonstrated the ability to use a StyleGAN approach on a very limited data set of fewer than 1000 highly diversified images of Pokémon characters and generate fairly convincing new characters with both images and names. Compared to traditional GAN approaches that require tens of thousands or even millions of training examples, this is a transformative size reduction that could be applied in many more applications and could be used to greatly advance animated character development for video games, movies, comics, and more.

References

[1] Chen J., Liu G., Chen X. (2020) AnimeGAN: A Novel Lightweight GAN for Photo Animation. In: Li K., Li W., Wang H., Liu Y. (eds) Artificial Intelligence Algorithms and Applications. ISICA 2019. Communications in Computer and Information Science, vol 1205. Springer, Singapore. https://doi.org/10.1007/978-981-15-5577-0_18

[2]Kleiber, J. (2020, August 10). PokéGAN: Generating Fake Pokémon with a Generative Adversarial Network. Retrieved from https://blog.jovian.ai/pokegan-generating-fake-pokemon-with-a-generative-adversarial-network-f540db81548d

[3] Lazarou, C. (2020, November 27). I Generated Thousands of New Pokémon using AI. Retrieved from https://towardsdatascience.com/i-generated-thousands-of-new-pokemon-using-ai-f8f09dc6477e

[4]Zhu, C. (2021, June 24). Teaching AI to Generate New Pokemon. Retrieved from https://m.mage.ai/teaching-ai-to-generate-new-pokemon-7ee0ac02c514

[5]Shimanuki, B. (2019). Joint generation of image and text with GANs (Doctoral dissertation, Massachusetts Institute of Technology).

[6] Pokémon Database. https://pokemondb.net/pokedex/national