ART AND MACHINE LEARNING
CMU 2022 SPRING
PROJECT 2

Manifesting Todd Bonzales

An image from a text-to-ai gan trained on a generalized dataset.
Made in early stages of experimentation.

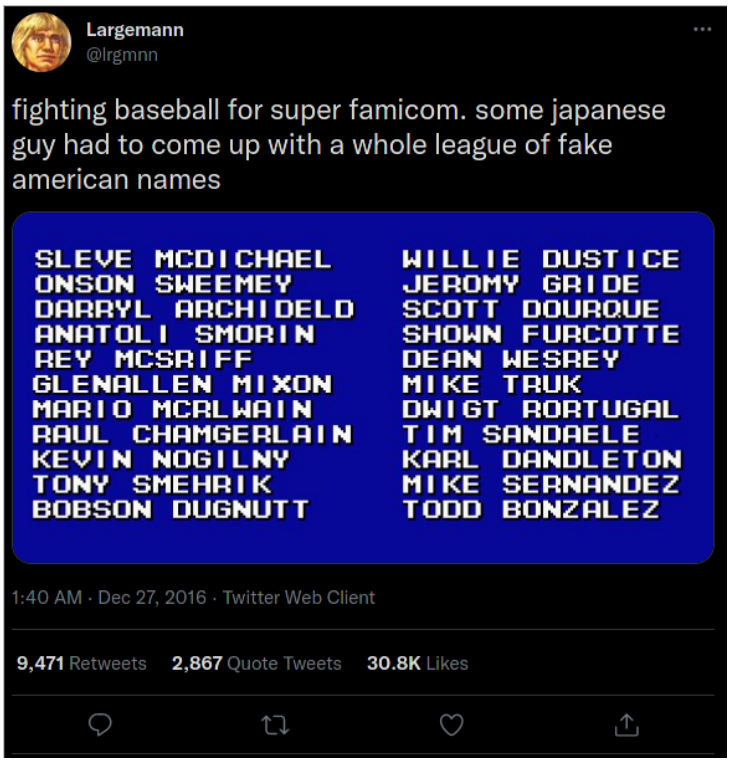
Will Rinkoff

DESCRIPTION

There were names of fictional baseball players that had no accompanying faces. I sought to use a text-to-image neural network to create some.

Concept

There was a viral tweet:



The japanese baseball game “Fighting Baseball” for the Super Famicom (released in 1995[1]) had to create an American team. However, the publisher had no deal with any teams, so they simply chose to make up a bunch of names that just *sounded* American, riffing on some existing American players. Of course, not much effort was made to make the names sound convincing, so what players got was a lineup of suspiciously off-kilter names, all of which are funny to both read and pronounce. I wanted to put faces to these names, hoping to canonically establish portraits for the players.

Technique

I wanted to take an existing text to image model (in this case, DALL-E), train it on a custom dataset of American baseball players, then ask it to make portraits for the fictional famicom lineup. I didn’t feel the project techniques were too complex. The outcome mostly hinged on the quality of the dataset, which I wasn’t overly concerned about because I found a fairly comprehensive set of players, each player having an identically-sized portrait image.

---

[1] https://thedragonfriends.fandom.com/wiki/Fighting_Baseball

Process

I wrote a small script to generate a dataset of baseball player names matched to urls with their pictures from the site 'baseball-reference.com', then I wrote another script that would (when needed) download all those players to a directory structure that could be loaded as a dataset into a pytorch implementation of DALL-E. This was the easy part, as I've had experience with web scraping. The *other* part was to train a DALL-E model to take player names and translate them to player portraits.

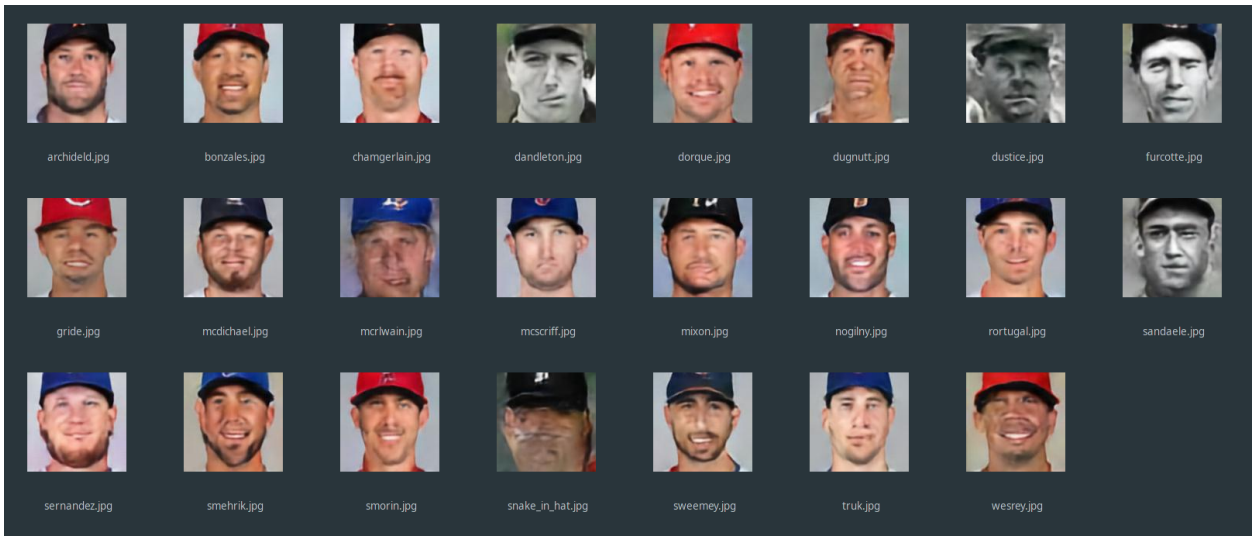
The first part I did in an afternoon. The second part I troubleshooted for some time, not knowing how to properly pass the data into the model, nor how to generate images for it. This was a major barrier for me until I actually went and read the README file. I realized there was a fairly robust cli for their implementation that only required a well-formatted dataset, which I had. I ran a few tests on google colab on small subsets of the data until I had composed a python notebook that worked. Now I went to amazon web services, booted up a server with gpus, and thought all I had to do was run the notebook and wait.

I spent an unreasonable amount of time trying to get the right versions of nvidia drivers, conda, pytorch, and cuda. I'm sure I consulted stack overflow 100+ times, each time running into a dependency issue at the first training step (the Variable Auto Encoder). At some point, I did get everything working (I had to use an aws environment catered towards machine learning, instead of the general art+ml 2022 environment). I spent a total of three days scraping, training, and generating. I generated 80 pictures for each player, and picked my favorite ones. Though I didn't have high hopes, I'm pretty happy with the results.

Reflection

If I were to do this again, I'd restrict the training set to pictures from years after the MLB had established standards for player portraits. Many of the older players didn't have portraits, and instead just had pictures of them on the field, which I think wasn't helpful for the results I wanted. Of course, I'd also make an effort to get familiar with the tools I'm using early in the project timeline.

RESULTS:



CODE

web scraping:

grab_mlb_players_v2.py: https://pastebin.com/raw/grXKVZcX

the dataset it generated:

https://drive.google.com/file/d/1t4qAXvXcWey0xLuzJD8dAkuI8rmNoM5p/view

get_imgs.py: https://pastebin.com/raw/19GAdprb

for ml:

ended up using the cli interface for this DALLE-pytorch implementation:

https://github.com/lucidrains/DALLE-pytorch