Pokemon Classification and Generation Using Convolutional Network and Generative Adversarial Network

Jingjie Ma
Georgetown University
Washington, DC
evelynjingjiema@gmail.com

Han Zhang
Georgetown University
Washington, DC
hz274@georgetown.edu

Abstract

Convolutional Neural Networks (CNNs) are widely used in image recognition. CNNs have the ability to learn different filters in a parallel fashion with respect to a training set. Generative Adversarial Networks (GANs) are used as a method for unsupervised learning. GANs will automatically discover useful patterns in the training data and generate outputs that are drawn from the training set plausibly (Creswell et al., 2017). In our project, we first classify the type and generation of Pokemon characters using Convolutional Neural Networks and then generate new Pokemon using Generative Adversarial Networks.

Introduction

Pokemon is a famous franchise that is centered on fictional creatures called "Pocket Monsters". In Pokemon games, players, as “trainers”, catch and train Pokemon to battle each other. There are over 800 Pokemon now after the release of the 8th generation game, and they all have unique names and types.

Pokemon can have single or dual types that enable them with different strengths and weaknesses. There are 18 types in total and the image below contains the names and representative colors of these types.

Figure 1. Types and Representative Colors of Pokemon

We want to examine if there exists any pattern in the design of Pokemon across different types and generations by performing classification on Pokemon images, and further try to generate new Pokemon using neural networks.

Methodology

Dataset and Pre-processing

There are three parts of data used in this project.

The first part is a CSV file that contains the name, type, and Pokedex (identification number) for each Pokemon. The second part is images of all Pokemon characters from generation 1 to 6. The last part is a CSV file containing the start and end Pokedex of each generation.

We merge the name, type, identification number with the image into one dataframe. In our project, we use 40% of the data as the training set, 30% of the data as the validation set, and the remaining 30% as the test set for all four of our models.

Pokemon Classification

Though Pokemon are fictional characters created by game designers, we have certain confidence in the models because according to the creators of Pokemon games, the inspiration of Pokemon characters are real-world animals (Loveridge, 2016). Charmander, a popular Pokemon in Figure 2 below, is a perfect example. This means that there might exist patterns in the colors and shapes of Pokemon across different types that could be learned by the models.

Figure 2. 004-Charmander (Pokemon) and a salamander

We construct a four-layer Convolutional Neural Network to classify the types of Pokemon. For each hidden layer, we use the rectifier function as the activation function. The output layer uses the softmax function as the output function. Adam is being used as the optimizer. Adam is a popular method that is used by stochastic gradient descent. Unlike regular gradient descent, stochastic gradient descent uses only a stochastic or random batch of training samples to update parameters.

Adam is a method that only requires first-order gradients with little memory requirement (Kingma & Ba, 2017). The method computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients The Adam optimizer does not only get stochastic gradient at each iteration, it also updates the first moment (the mean) and the second raw moment (the uncentered variance) accordingly (Kingma & Ba, 2017). The new parameter will be updated based on these two moments (Kingma & Ba, 2017).

Since the dataset contains images of Pokemon from different generations, we also train a model to classify the generation of a Pokemon. Since there are over a hundred Pokemon in each generation, and every generation includes several games released in a certain period, we expect differences in the artistic styles between generations. The theme and designers change over time, so it is possible to discover patterns in the design of Pokemon over different generations.

A four-layer Convolutional Neural Network is used here. Similar to the CNN model for type classification, we are using the same set of hidden layers. However, the activation function of the generation classification model is the Sigmoid function.

Generating New Pokemon

We use Generative Adversarial Networks to generate new Pokemon characters. Generative Adversarial Network is an automatic process that both learns and generates new outputs (Wang). A GAN model typically consists of two parts, a generative model and a discriminative model.

In our project, specifically, the generative model generates new "fake" images of pokemon characters so that the discriminative model can "evaluate" those images. In particular, the generative model will learn the distribution of the training set. The discriminative model gives a binary output, in other words, it predicts whether the input image is actually from the training set or is generated by the generative model. The goal of the generative model is trying to increase the error rate of the discriminative model.

The learning process of the generative model starts at some specific latent space. In our model, we use multivariate normal distribution as the latent space. We train the data from the original normal distribution to the distribution of Pokemon. The generative model is learning the

distribution of the training set. Thus, the generative model generates a distribution of pixels that is similar to the distribution of Pokemon pixels.

We use a three-layer Convolutional Neural Network as the generative network. For each hidden layer, we use the Leaky ReLU function as the activation function and the Sigmoid function as the output function. A six-layer Convolutional Neural Network is used as the discriminative model. Similar to the discriminator, we use the Leaky ReLU function as the activation function and the sigmoid function as the output function. We train the discriminator by mixing fake Pokemon and real Pokemon.

When training the generator, we fix the weights of the discriminator, and vice versa, to avoid the error of backpropagation the whole system. Thus, the overall classification error is maximized after each iteration of the generator training because the generator is trying to generate satisfiable "fake" Pokemon that look like the real ones so that the discriminator cannot tell the difference. The overall classification error is minimized after one round of training the discriminator so that it can help improve the accuracy of the generator in the next iteration.

Results and Model Evaluation

Pokemon Classification

We first train the model with Pokemon from four different types, namely, fire, water, grass, and electrics. Typically, Pokemon from these four types would have some characteristics from certain animals, plants, or some natural phenomena. For example, some fire Pokemon are red and might have fire shape design on them. Thus, we can expect that the fire Pokemon tend to

be red, the water Pokemon tend to be blue, the grass Pokemon tend to be green, and the electric Pokemon tend to be yellow.

Figure 3. 005-Charmeleon (Fire), 230-Kingdra (Water), 001-Bulbasaur (Grass) and 172-Pichu (Electric)

Since we only have one image for each Pokemon, in other words, we only have 721 images in total, our test set is highly imbalanced. Thus, we are not using the test accuracy as the evaluation metrics for all type classification models. Instead, we use the accuracy rate on the validation set for model evaluation purposes.

As shown in Figure 4 below, our model achieves about 40% accuracy on the validation set. The model ignores the shape of Pokemon but rather captures the color of them.

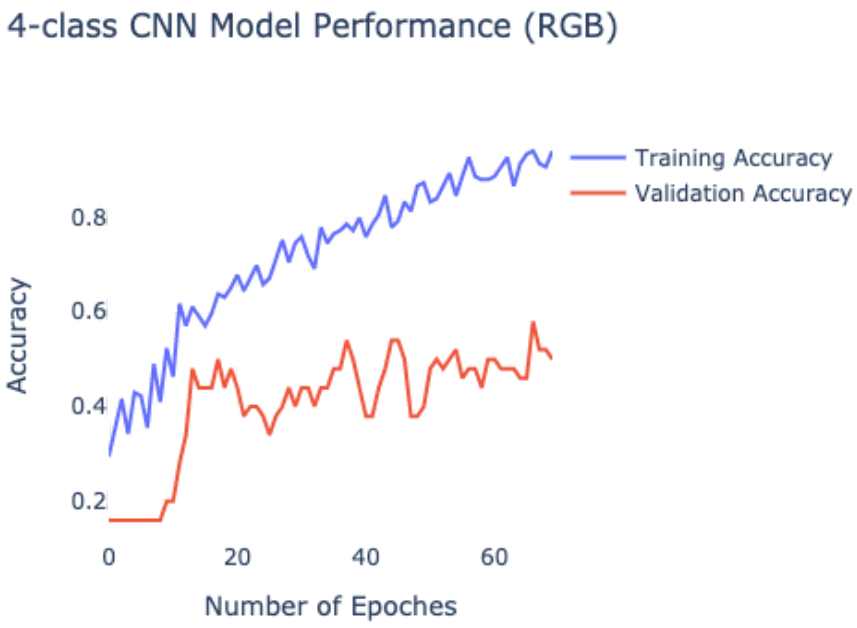

Figure 4. Accuracy of the CNN with four classes (Fire, Water, Grass and Electric)

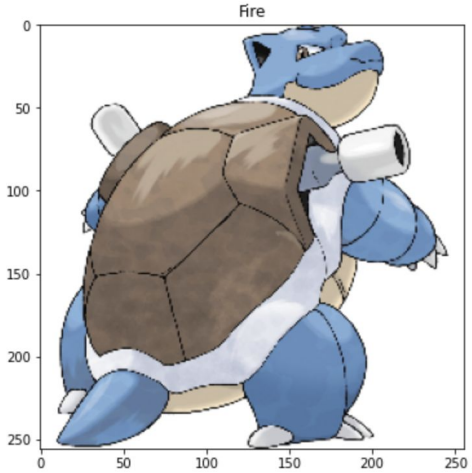

Figure 5. 009-Blastoise (Water), Misclassified by the 4-class model

For example, the Pokemon in Figure 5 is of water type, but it is classified as type fire. Since it has a brown shell, the image will have higher red and green value.

We then train the model using three types of Pokemon: water, fire, and grass. Intuitively, images of water Pokemon should be high in red value. Images of water Pokemon should be high in blue value and the images of grass Pokemon should be high in green value. Figure 6 shows the accuracy of the model on both the training set and the validation set.

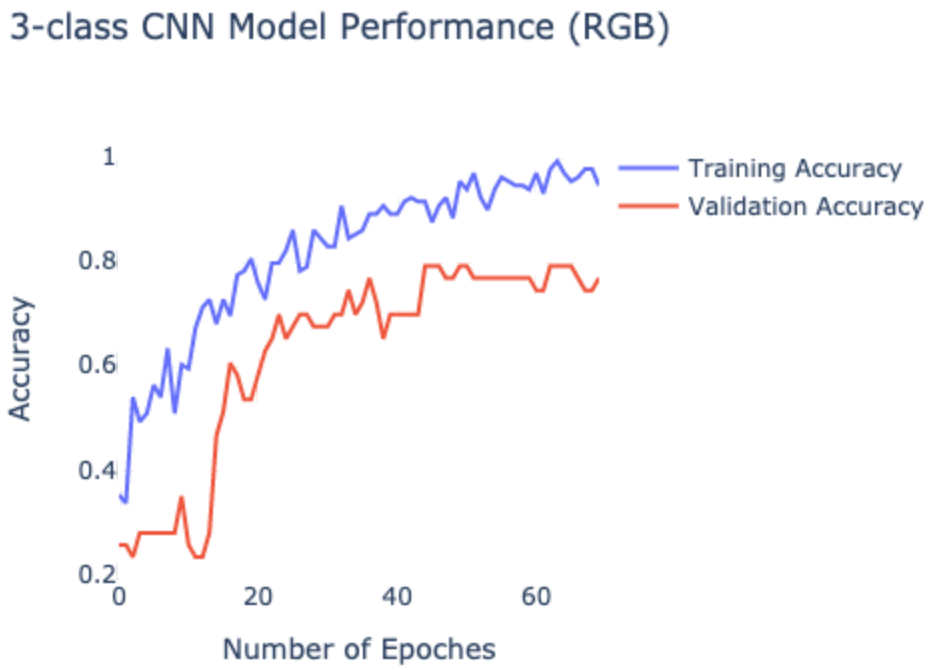

Figure 6. Accuracy of the CNN with three classes (Water, Fire, and Grass)

The model that classifies three classes performs much better than the first one. The accuracy of the model on the validation set is much higher. One potential reason for the result is that the RGB value of a yellow pixel will have both green and red values. For example, the color yellow, with the Hex code #FFFF00, has green value 255, red value 255, and blue value 0. Again, we find that our model captures the differences in color rather than the differences in shape. Thus, our model could be useful to classify classes that are different in terms of color scheme.

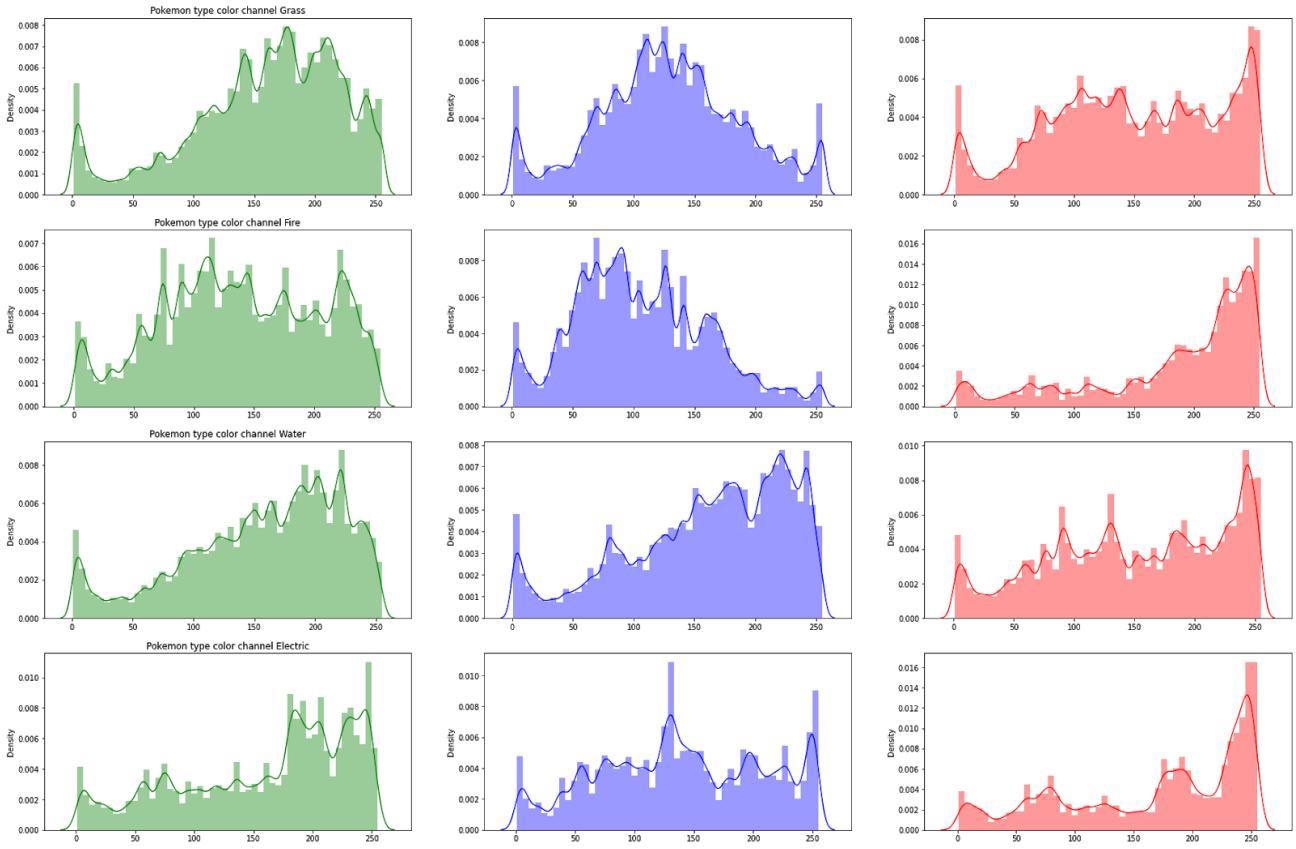

Figure 7. The distribution of RGB Value of Type Grass, Fire, Water and Electric

By looking at the distribution of RGB channels for the four types of Pokemon (grass, fire, water and electric), we realized that the water and the fire type is the most distinguishable pair among all combinations. The accuracy plot is shown below in Figure 8.

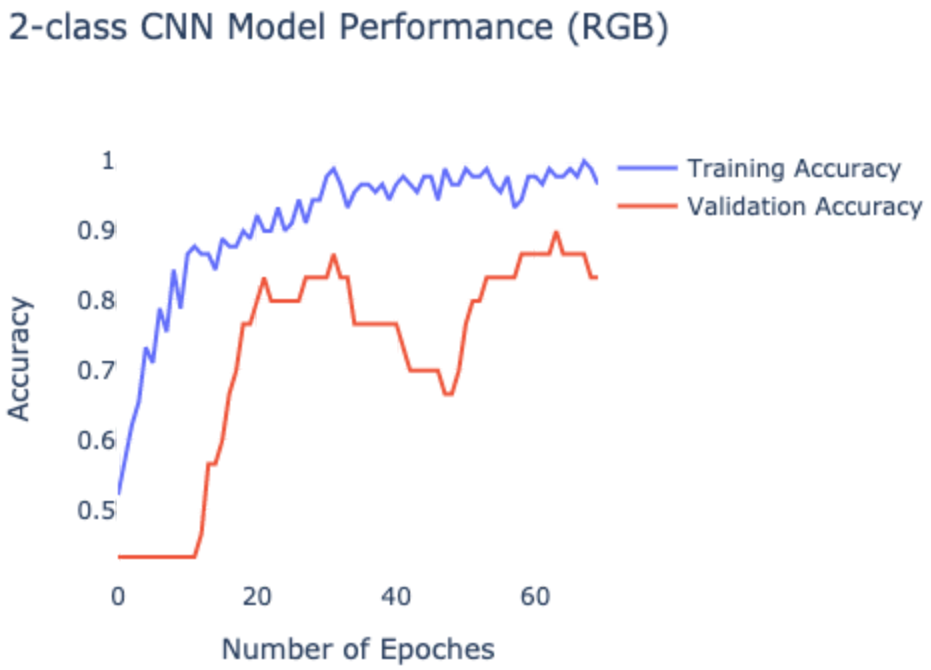

Figure 8. Accuracy of the CNN with two classes (Water and Fire)

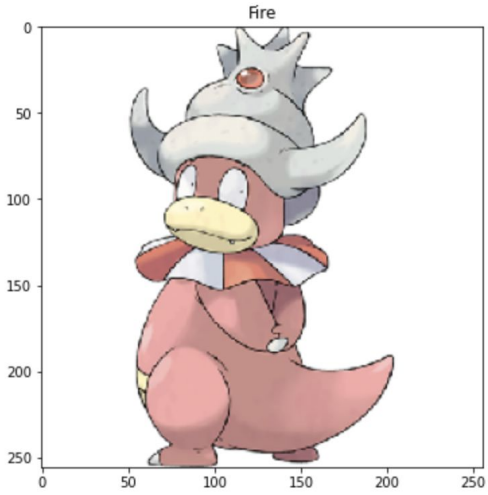

Figure 9. 199-Slowking (Water), Misclassified by the 2-class model

Though the model misclassified Pokemon when the character does not follow the typical color scheme of the Pokemon world (as shown in Figure 9 above), the model achieves nearly

90% accuracy. Thus, we can conclude that our model will perform better when the input is drastically different in terms of color schemes.

Overall, our CNN model yields satisfactory results to identify the types of the Pokemon, especially when the input Pokemon differ in color schemes. Our model captures the differences in color but does not distinguish Pokemon types based on the shape.

In order to confirm that it is not feasible to classify based solely on the shape of Pokemon, we conduct the same set of models using the gray scale images of Pokemon. The following plots show the accuracy of the models with gray scale inputs.

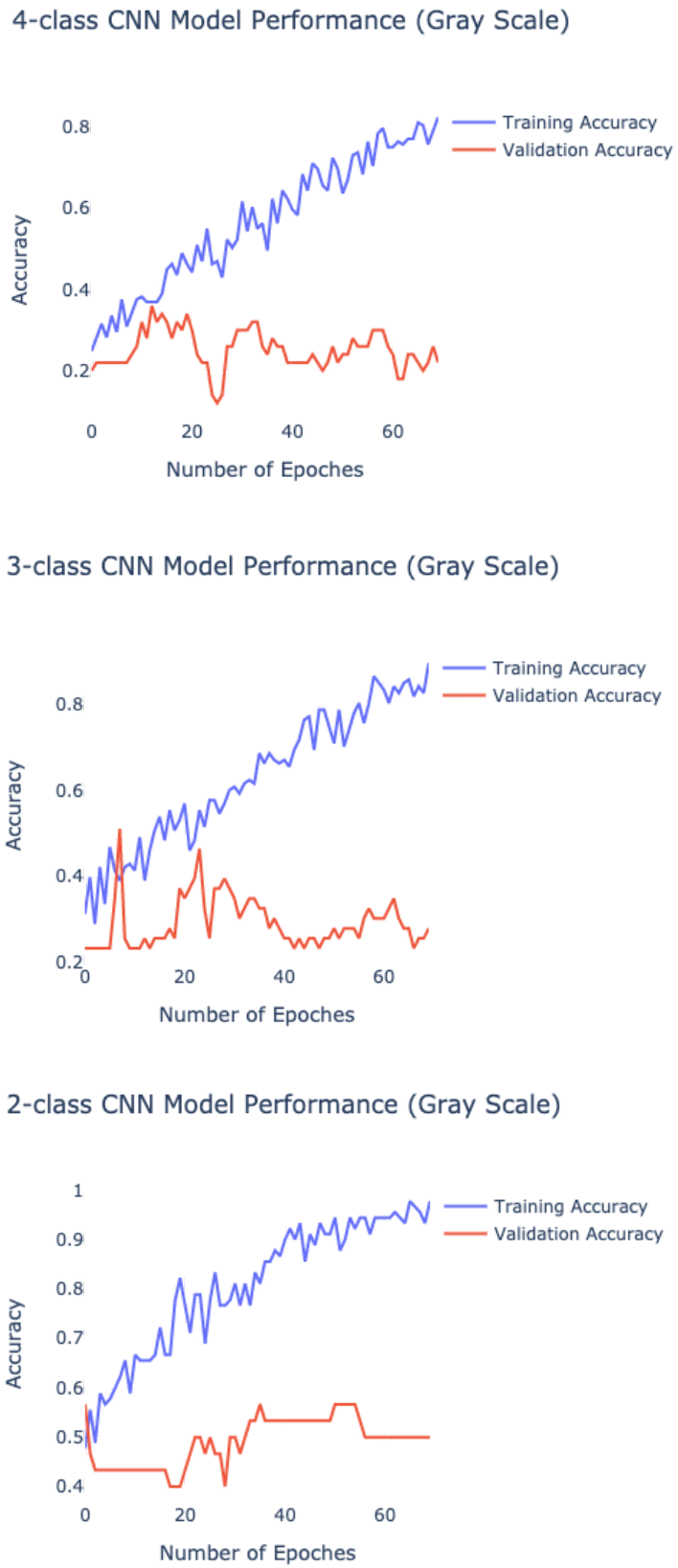

Figure 10. Accuracy of the CNN models with 4/3/2 classes (gray scale)

As we can see in the plots above, our model cannot successfully classify the type of Pokemon using gray scale images. The reasons could be two-fold.

First, Pokemon characters tend to follow characteristics of some real-life animals and plants. This is a double-edged sword. The reason why our model works partially is that Pokemon characters of the same type share common features, for example, grass Pokemon mimicking green plants and water Pokemon mimicking sea animals. Thus, our model could classify Pokemon based on color. However, even those "zoosemy" Pokemon have different taxonomy in biology. For example, fish-like and crab-like Pokemon can both be water types in the Pokemon world, however, they are different species biologically and in fact quite divergent in terms of appearances. So although some Pokemon characters belong to the same type, their shape could be completely different. Moreover, as the worldview expands, the design of Pokemon becomes more complex and rich in details. Designers also make Pokemon mimicking all sorts of real-life and imaginary objects, as shown in Figure 11 below, which could make it hard for the model to learn from the shape. Thus, our model mainly captures the color scheme of each type.

Figure 11. 154-Meganium (Grass) vs 455-Carnivine (Grass) vs 652-Chesnaught (Grass) and 179-Mareep (Electric) vs 479-Rotom (Electric) vs 603-Eelektrik (Electric)

Also, we only have one image for each Pokemon. Therefore, all Pokemon characters are two-dimensional. However, these Pokemon characters are three-dimensional by design. Thus, our model is not able to capture enough shape differences for classification.

Since the number of Pokemon in each generation has a roughly uniform distribution, we use the test accuracy to evaluate the CNN model that classifies the generation of a Pokemon. Our model achieves 83% accuracy and the input has six classes. This high accuracy is reasonable since every generation of Pokemon games has its own worldview, and designers need to fit Pokemon characters into that environment. Just like what the director Shigeru Ohmori said: "... perhaps we're making a Pokémon that's going to appear in a cave, so perhaps it would fit well with that location" (Loveridge, 2016). This guideline for designers allows Pokemon in the same generation to share more common characteristics, and thus enable the model with such high accuracy when predicting the generation to which a Pokemon belongs.

Generating New Pokemon

Though we train our model using GAN and GAN tends to yield satisfactory results for generating human faces, we do not get "fake" Pokemon images that look identical to real Pokemon, as shown in Figure 11. However, the images we generated have the vague shapes of Pokemon.

One reason for this is that Pokemon are created by human imagination. Technically, they can be derived from any object or they could be purely imaginary. There is no solid rule of how Pokemon characters should look like. Also, since each generation of game has its own "game universe", and Pokemon need to fit in that specific environment, the Pokemon images we used,

which contains characters from 6 different generations, may not have many features in common. Thus, generating Pokemon characters may not be as easy as generating human faces.

Also, due to the fact that our sample size is relatively small, our model may not be able to accurately capture important features of Pokemon.

Figure 11. GAN Generated Pokemon

Limitations and Recommendations for future Researches

The limitation in our model is that we only do Pokemon classification on certain types. This is due to the fact that Pokemon of type fire, water, grass, and electrics are more regular and predictable than those of other types. Unlike these Pokemon, ghost, fairy and psychic Pokemon are more spiritual and vary a lot in terms of shape and color. Poison, dragon, ground and bug Pokemon are more imaginary instead of realistic, making them also hard to learn.

There are also limitations in our dataset. The first one is that our dataset contains only one two-dimensional image for each Pokemon. Pokemon are characters in 3D. The color and the shape of a Pokemon could be different from different angles. Thus, we recommend future researchers who are interested in this topic to collect multiple images for each character. Similar to face recognition, where facial expression might influence the accuracy of the model, we highly recommend collecting Pokemon with different gestures. It could be the case that our model is always misclassifying Pokemon in certain gestures. Also, we recommend future research to construct models based on the 3D representations of Pokemon, if possible.

Conclusions and Thoughts

We can see that the CNN models yield satisfactory results in identification of types of Pokemon, especially when the input Pokemon differ in color schemes. Our model captures the differences in color but does not distinguish Pokemon types based on the shape. This is mainly because the design tends to follow characteristics of some real-life animals and plants, but only to a certain extent. Designers always adjust the characters based on the environment and use their

imagination to make the appearances of Pokemon more attractive and distinguishable. So the models learn better from the colors of Pokemon rather than shapes. Also, since Pokemon characters from the same generation share common features, our CNN model works well when predicting which generation a Pokemon belongs to.

Pokemon generation shows that human creativity has no limits. Though inspired by real-life animals and plants, Pokemon designers create multiple universes and characters that do not originate in our daily lives. Therefore, the more we learn about deep learning, the more we appreciate humanity. Nearly everything we value today comes from imagination. There will be no money, no human rights, even no nations without the collective imagination of human beings (Harari, 2015). Though artificial intelligence has made our lives much easier than ever before, please remember to be creative and never stop imagining.

References

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. (2017, October 19). Generative Adversarial Networks: An Overview. Retrieved December 15, 2020, from https://arxiv.org/abs/1710.07035

Harari, Y. N. (2015). *Sapiens: A brief history of humankind*. New York: Harper Perennial.

Kingma, D., & Ba, J. (2017, January 30). Adam: A Method for Stochastic Optimization. Retrieved December 15, 2020, from https://arxiv.org/abs/1412.6980

Loveridge, S. (2016, October 20). Want to know how The Pokémon Company designs Pokémon? Retrieved December 16, 2020, from https://www.digitalspy.com/videogames/pokemon/a811377/heres-how-the-pokemon-company-design-pokemon/

Wang, K. (n.d.). Generative adversarial networks: Introduction and outlook. Retrieved December 15, 2020, from https://ieeexplore.ieee.org/abstract/document/8039016