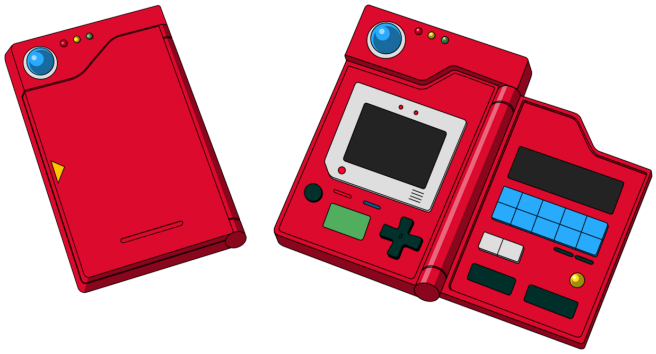
I. Definition

Project Overview

This project's objective is to build a image classification program akin to the Pokédex idea from the Pokémon franchise, in that it will input an image of a Pokémon (from a set of 20) and correctly identify with a label which Pokémon the observer has input. The project was inspired by my interest and hobby of video games and in particular my fondness to the Pokémon universe.



Problem Statement

To identify by name the Pokémon that has been input from an image of that Pokémon. This will be solved by the use of a Convolutional Neural Network trained on images collected from the internet. The images will be augmented and synthetically created to expand the data set so that each of the 20 classes are properly represented.

The steps for the project are as follows:

1. Create a web scraping tool to take images from the internet of each of the 20 Pokémon selected
2. Organise those images into training and validation sets
3. Augment the images to expand the number of images that are being used to train the CNN
4. Train the CNN and tune hyperparameters
5. Evaluate performance of the model
6. Predict on some unseen images

Metrics

A simple accuracy measure would be how many times a Pokémon is correctly identified.
A Confusion Matrix is also important to understand the precision and recall of the machine learning algorithm.
The output can be measured by using an assessment methods such as accuracy, recall, precision and F1 score. These will be used to alleviate what is known as the Accuracy Paradox.

II. Analysis

Data Exploration

In this section, you will be expected to analyze the data you are using for the problem. This data can either be in the form of a dataset (or datasets), input data (or input files), or even an environment. The type of data should be thoroughly described and, if possible, have basic statistics and information presented (such as discussion of input features or defining characteristics about the input or environment). Any abnormalities or interesting qualities about the data that may need to be addressed have been identified (such as features that need to be transformed or the possibility of outliers). Questions to ask yourself when writing this section:
- _If a dataset is present for this problem, have you thoroughly discussed certain features about the dataset? Has a data sample been provided to the reader?_
- _If a dataset is present for this problem, are statistics about the dataset calculated and reported? Have any relevant results from this calculation been discussed?_
- _If a dataset is **not** present for this problem, has discussion been made about the input space or input data for your problem?_
- _Are there any abnormalities or characteristics about the input space or dataset that need to be addressed? (categorical variables, missing values, outliers, etc.)_

Exploratory Visualization

Here are a few images that the ConvNet will be trained on. These are the types of images prior to augmentation.

Algorithms and Techniques

The classifier is a Convolutional Neural Network (ConvNet), which is an algorithm for most image processing tasks, including classification. The algorithm
outputs an assigned probability for each class and the following parameters can be tuned to optimize the classifier:

- Classification threshold
- Training parameters
- Training length (number of epochs)
- Batch size (how many images to look at once during a single training step)
- Solver type (what algorithm to use for learning)
- Learning rate (how fast to learn; this can be dynamic)
- Weight decay (prevents the model being dominated by a few “neurons”)
- Momentum (takes the previous learning step into account when calculating the next one)

Neural network architecture:

- Number of layers
- Layer types (convolutional, fully-connected, or pooling)

- Layer parameters

During training, both the training and the validation sets are loaded into the RAM. After that, random batches are selected to be loaded into the GPU memory for processing. The training is done using the Mini-batch gradient descent algorithm (with momentum).

Benchmark

The benchmark model will be a simple random selector of Pokémon recognition with no learning, ie. the name of the Pokémon is picked at random from the twenty different names available. This will be measurable and can be used to benchmark the learning algorithm. A dummy classifier will be used from *sci-kit learn*.

III. Methodology

Data Preprocessing

Once the images were collected by the collector program that scrapes the web for the images, the images needed to be resized and alpha removed so that the channels were correct for Keras. They would then need to be augmented using the *DataImageGenerator* function in keras that allows for the augmentation of images on the fly.

The images also needed to be normalised so that each channel of the image is a number between 0 and 1 and finally, Keras requires to specify the input shape of the image so all images need to be of the same size.

Implementation

Refinement

The following adjustments were made to the model during training:

1. Number of feature maps in Convolutional Layers
2. Dimensions of feature maps in Convolutional Layers
3. Dimensions of pooling matrices in Pooling layers
4. Number of hidden units in layer between the feature extractor and the output layer
5. Epoch sized
6. Batch size
7. Dropout rate
8.

These were all optimised using the *hyperas* package which works alongside *keras* to tune the hyperparameters and optimise the model that is being trained.

4. Results

Model Evaluation and Validation

Overall accuracy of the validation set of images was 73% which is suitable for this project, this level is also continued when one looks at the F1 score which is also high at 72.5%.

The metric score for 'Jolteon' images is particularly poor in comparison to the other images. I am unsure why this might be the case. 'Jigglypuff' however is easily the most successfully modelled Pokémon, perhaps due to its distinctively round shape.

Model accuracy slow but steadily increased with every epoch but began to plateau at around 8 epochs on the training data.
Justification

The model generated by the ConvNet produces higher accuracy than the Dummy Classifier model showing that it is indeed a better classifier than a benchmark model.

5. Conclusion

Free-Form Visualization

Reflection

Overall I am pleased with what I have produced as this has been my first time using Keras and ConvNets. The difficulty that I faced was with coding in Python and manipulating image files into numpy arrays as I had never done this before. The project took much longer than anticipated but I found the use of Python for this project has really improved my confidence in the language and I feel more ready to tackle more traditional machine learning problems now that I have worked with a neural network.

Improvement

The project could have been improved by creating a much deeper model and having access to thousands of images in each class to train on. With an accuracy of ~73% already on validation data I feel like this could be pushed to over 90% if these improvements were implemented.

by Lee Joshi-Jones