ShinyTch a shiny hunting tool

A study into what the most efficient way of counting when hunting shiny Pokemon is

Shane Lawson

University of Lincoln, School of Computer Science

18680054@students.lincoln.ac.uk

Word Count : 1,193

November 16, 2021

There are many websites and applications made to track people progress of hunting shiny Pokémon, these all use keyboard presses to track the amount of Pokémon that has been encountered, this study is to see if automatically tracking when you encounter a Pokémon saves time when hunting for shiny Pokémon

Keywords: Machine Learning, TensorFlow, Pokemon Counter.

1 Introduction

“Pokémon are creatures of all shapes and sizes who live in the wild or alongside humans” (Pokémon, 1996) Shiny Pokémon are available in most Pokémon games and have a rare chance of appearing, most of the time taking hours of time to hunt for, with the task being so repetitive its often hard to focus on something else while hunting shiny Pokémon without risking accidently losing the shiny Pokémon, my goal is to create a program that automatically counts each Pokémon the encountered and play a sound of choice when the shiny Pokémon is encountered

I plan to try using two different ways of detecting the Pokémon, this is going to be done by using reading the color of a selected section on the capture cards output, which can be selected to allow for different resolutions and different Pokémon games, a issue with this is that its not very accurate due to some Pokémon having the same colors, This is not an issue for the second way of detecting Pokémon is using TensorFlow (Google Brain, 2015) which is an open source software to detect what the Pokémon is and if it is shiny, this will be slower but has a higher accuracy and the ability of detecting other shiny Pokémon than the one selected for the hunt

The different methods will be compared with pressing a button on a keyboard each time a Pokémon is encountered, the time between encountering Pokémon will be tracked to see how efficient each one is for getting most encounters per hour. Each of the different methods were inspired by a YouTube (Google, 2005) video posted by creator Nooby (Nooby, 2014) in which they use a camera to detect the rgb value of the Pokémon on screen. The other two methods were inspired by another YouTube video from another creator Livingmaniac (Livingmaniac, 2021) which uses a capture card and ai to detect which Pokémon is on the screen and if it is shiny and the last is the keyboard-based counter like the *website* ShinyHunt (PokétchApp 2019).

2 Aims and Objectives

The aim of this project is to see which method of tracking shiny Pokémon is the most efficient for getting the most encounters per hour, this is going to be done by tracking the time between each encounter, for the keyboard tracker this will include the time it takes the user to press the keyboard. With a collection of completed hunts by the participants in the study I will be finding on average what method had the most encounters per counting method depending on how the participant chose to hunt

During the study I will conduct the following:

- Test which coding language would be better for the project, changing the software used from TensorFlow to a software that implements the complete TensorFlow API in C# called TensorFlow.NET (SciSharp. 2019)
- Train TensorFlow to be able to recognise each Pokémon
- Tracking the time spent between encounters and the total time spent hunting the shiny Pokemon, this will also be stored with the Pokemon that is caught
- Store the Pokémon the user has hunted and caught using the application, this data will also be shown in a list on a separate section of the application
- A conclusion and verdict will be reached on what the most efficient way of counting encounters when hunting shiny Pokemon.

3 Academic Literature

I found an academic paper where TensorFlow is used to classify flowers, they also compare multiple different architecture width multipliers to find the which resulted in better accuracy, In the paper they also compare the resolution of the images used to train TensorFlow with smaller resolution images being used a decrease in accuracy is noticed. This will be important for helping me achieve the highest accuracy (N. R. Gavai, Y. A. Jakhade, S. A. Tribhuvan and R. Bhattad. 2017) I also found another paper that test some other architectures to determine which one has the best accuracy (Zoph B, Vasudevan V, Shlens J, Le Q. V. 2018).

The second paper I found was again about TensorFlow, the paper talks about using a image database called ImageNet (ImageNet. 2018) which provides TensorFlow with the data needed to train itself on being more accurate on detecting what’s in the images, this database has hundreds of images for multiple different subjects to train on, for example this paper also talks about flowers (Jubin Dipakkumar Kothari. 2018)

I found a paper where they were able use both TensorFlow and ImageNet to get a accuracy of 99.823% on one of their test flowers, this will be extremely useful when training TensorFlow for my project, this is because of there being multiple Pokémon, each Pokemon will require over 400 images each, this is to provide the highest accuracy possibly (Abu, M.A & Indra, N.H & Abd Rahman, A & Sapiee, N & Ahmad, I. 2019)

This paper has some methods of determining the RGB values of a pixel, this will be useful for determining the main rgb value of the encountered Pokemon, I will need to use this sort of method to compare it against a database of all the correct colors for a shiny to be registered (Vezhnevets V, Sazonov V, Andreeva A. 2003). From the RGB value I will be able to get a hex color that I can then mat with a database I will make for each Pokémon’s shiny hex color, this database will also include the hex color of each Pokémon’s normal hex color

4 Project Plan and Risk Analysis

Im going to be planning the project using a gantt chart, this includes the time already spent making this proposal, including the testing and final report. I have chosen the times spent on each part of the project on how difficult I see each part being and allowing for being ahead or behind,

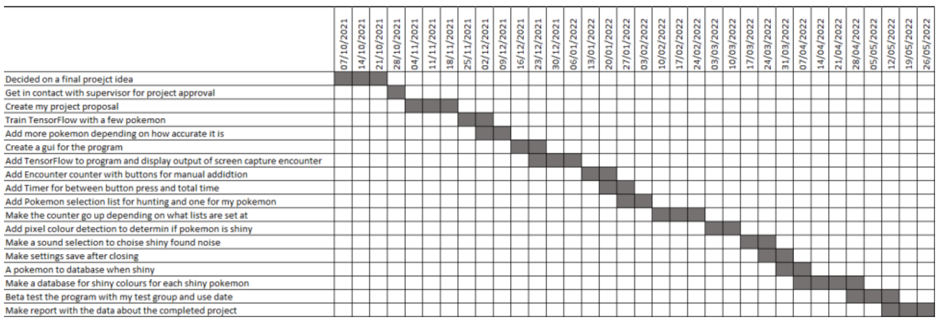

Fig. 1. My Gantt chart that shows my predicted time for project parts.

4.1 Risk Analysis

Risk No.	Specific Risk Details	Likelihood	Assessed Impact	Manage/Mitigation Action
1	The open-source software TensorFlow becoming proprietary software	Low	High	Make sure that there is other software I can use instead in case TensorFlow does become proprietary software
2	Government policies change / ethical guidelines change	Low	Medium	Due to what my project is about it will most likely not affect it, but it would be changed to fit the new guidelines or policies
3	Takes to long to add all Pokemon to the program	Medium	High	Although this would limit the program, I would start by adding the Pokemon that are available on the latest games

4	Participant’s not encountering a shiny Pokémon before the project is completed	Medium	Medium	Select participants that are likely to spent more time to get a shiny instead of participants that would likely not finish

References

Pokémon (1996) *The Official Pokémon Website*. Japan: The Pokémon Company Available at: https://www.pokemon.com/uk/parents-guide [accessed 16 Nov. 2021].

Google Brain (2015) *TensorFlow* [Software]. Google Brain Available at: https://www.tensorflow.org [accessed 16 Nov. 2021].

Google (2005) *YouTube*. San Mateo: Jawed Karim, Chad Hurley, Steve Chen. Available from https://www.youtube.com/ [accessed 16 Nov. 2021].

Nooby (2014) *Automatic Soft Resetting Shiny Pokemon Finder (Shiny Mewtwo Encounter*) [Video]. Available from: https://www.youtube.com/watch?v=E9rO9NdW49o [accessed 16 Nov. 2021].

Livingmaniac (2021) *Auto Shiny Pokemon Counter v2* [Video]. Available from: https://www.youtube.com/watch?v=BIfolVN2TRQ [accessed 16 Nov. 2021].

PokétchApp (2019) *Pokétch ShinyHunt* PokétchApp. Available from: https://shinyhunt.com/ [accessed 16 Nov. 2021].

SciSharp (2019) TensorFlow.NET [Software]. SciSharp. Available from: https://github.com/SciSharp/TensorFlow.NET [accessed 16 Nov. 2021]

ImageNet (2018) ImageNet Object Localization Challenge | Kaggle. ImageNet. Available from: https://www.kaggle.com/c/imagenet-object-localization-challenge/overview/description [accessed 16 Nov. 2021]

N. R. Gavai, Y. A. Jakhade, S. A. Tribhuvan and R. Bhattad, (2017) *MobileNets for flower classification using TensorFlow*, International Conference on Big Data, IoT and Data Science (BID). Available from: https://ieeexplore.ieee.org/abstract/document/8336590 [accessed 16 Nov. 2021]

Jubin Dipakkumar Kothari (2018). *A Case Study of Image Classification Based on Deep Learning Using TensorFlow*. International Journal of Innovative Research in Computer and Communication Engineering. Available from: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3729757 [accessed 16 Nov. 2021]

Abu, M.A & Indra, N.H & Abd Rahman, A & Sapiee, N & Ahmad, I. (2019). *A study on Image Classification based on Deep Learning and Tensorflow*. Universiti Teknologi Malaysia, R.T Technology Sdn Bhd, University of Kuala Lumpur. Available from: https://www.researchgate.net/profile/Mohd-Azlan-Abu/publication/332850035_A_study_on_Image_Classification_based_on_Deep_Learning_and_Tensorflow/links/5cccd2dfa6fdccc9dd8b3e69/A-study-on-Image-Classification-based-on-Deep-Learning-and-Tensorflow.pdf [accessed 16 Nov. 2021]

Zoph B, Vasudevan V, Shlens J, Le Q. V (2018). *Learning Transferable Architectures for Scalable Image Recognition*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Available from: https://openaccess.thecvf.com/content_cvpr_2018/html/Zoph_Learning_Transferable_Architectures_CVPR_2018_paper.html [accessed 16 Nov. 2021]

Vezhnevets V, Sazonov V, Andreeva A (2003). A Survey on Pixel-Based Skin Color Detection Techniques. Moscow State University. Available from: https://d1wqtxts1xzle7.cloudfront.net/33285151/2003_-_Vezhnevets__Sazonov__Andreeva_-_A_Survey_on_Pixel-Based_Skin_Color_Detection_Techniques_-_ICCGV-with-cover-page-v2.pdf?Expires=1637193258&Signature=OZ-luc2Mj4JTeGmSgBxv6N4UPfaJyx9v5sWhKZzrVhAuqmxlss67qcxI1asG~NGC0Uef3uXCcIo~1EluFTdhA0KCpOMumjKSL3VWZvqZR5BAmFOBONyXWgx49jx70ukbuFZ-3KLqlmA2U4VGPbxKN1hQWguCHHdzCMbSnl-m2N6WJDpn-MuCJg1VkuetcD4JtTtKDV0~6sYhOBwjXFZU7EyiG~DaJrssmcBZjPsEUoDBMRHN-FE-PUq7JPOuzB6JSjNkAEsPHxtXxJ1Ywkfx6uMozXoxA2SStXre0PYVaNrxFng4Y1AmATXdaipOLBi5SeacxW~M0WTSDjmd2OUfhw__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA [accessed 16 Nov. 2021]

Based_Text_Representation_for_Document_Classification [accessed 03 Nov. 2020].