An Intelligent Drone-Based Hierarchical Image Classifier: Defense against Mosquito Borne Illnesses

Abstract

With a continual threat of mosquitoes carrying deadly diseases such as Zika, Malaria, and Yellow Fever, we as humans have constantly kept our guard up through the use of medical technologies such as vaccines, that have decreased our chances of contracting a disease from these mosquitoes. Why can't we, instead, head straight to the source of these diseases, and find out where these mosquitoes are breeding? By attacking them at their breeding grounds, we can control their diffusion to more populated areas to keep the diseases from spreading along with them. Through the use of a drone-based system that has been trained to recognize mosquitoes and mosquito larvae at their breeding grounds, we can deploy small pellets or a powdered form of Bti, an organic and safe insecticide, to kill off the mosquitoes. Through this process, not only can we decrease the population of mosquitoes, but also the population of those carrying a disease, lowering the chances of contracting one by a large amount. My proposal is to employ Artificial Neural Networks, specifically Convolutional Neural Networks, to train a drone offline, using Google's Machine Learning framework Tensorflow, aided by GPU acceleration. My hope is that with this approach, aided by the new technological advancements of today, we can now target very point-specific locations rather than large general areas of mosquitoes to help solve the problem of diseases running rampant worldwide.

Background Research

I have looked into previous efforts made by humans to exterminate mosquitoes. Some include using human operated crop duster to drop insecticide over these areas. In addition, people have employed the use of fish that eat the larvae. However, the human intervention aspect of physically transporting the fish from one location to another is very cumbersome and not efficient. I have also researched into my particular problem of recognizing these mosquitoes and mosquito larvae. I have learned that since this is a classification problem, and this problem can be solved using logistic regression, I'm looking into softmax and gradient descent regressions as a more specific solution. From this point, I learned of the software library Tensorflow, from Google, which could aid me in creating a deep neural network for image classification.

Challenges with Autonomous Drone based Approach

- Having drone navigate through a myriad of different environments and obstacles. In other words giving the drone complete autonomous capabilities.
- **Effectively locating stagnant water where mosquito larvae may be present.**
- Delivering payload based on strain on drone, and distance drone is able to travel.
- **Developing an efficient scanning method to locate collections of stagnant water, while taking into account the compute capability of the drone to control the speed at which the drone can continuously move**.
- Assessing a reasonable height at which the drone can capture all necessary features for image classification.
- Training Deep Neural Network as to have the greatest accuracy while sacrificing the least possible compute power.

Specific Problem I’m Focusing On

- The particular area of this problem I’m focusing on is the actual classification of the images, and the algorithm behind it that will allow this process to proceed smoothly.
  - This algorithm essentially segments an image as to amplify the features that are of importance, such as stagnant waterbodies.
- By finding an effective solution to this problem, I will be able to start new experimentation that will deal with this program’s compatibility with the live feed the drone will be giving the classifier.
  - A possible means of doing this is by modifying the drone to become compatible with something like a Jetson TK1/TX1 that has camera support. This will allow for the pre-trained Deep Learning Model to be uploaded to the drone, and then it will be able to classify as it is flying, allowing it to efficiently exterminate mosquitoes.

Hypothesis

In order to solve this problem, a machine learning approach can be used, primarily by utilizing a deep neural network. This neural network should be trained upon a large enough set of images to produce the accuracy required to distinguish between the classes. The two main problems that need to be solved are that the drone needs to locate the stagnant water where mosquitoes tend to breed, and the drone needs to then locate the mosquitoes and the mosquito larvae themselves. Using Machine Learning techniques, specifically image recognition, I should be able to approach these two problems through a two-step process covered by a single program. This program should work by taking a raw input image and finding the highest weighted sum of first the waterbody, based on its previous training, and then the mosquitoes.

Solutions for Breeding Grounds

- After some initial research, I found that to fight the problem of mosquito-born illnesses, researches have employed the use of fish such as, goldfish, mosquito fish, minnows, and koi to eat mosquito larvae.
  - Although this seems to effective, it requires a human to physically transport the fish from place to place, which is very slow and inefficient.
- Eventually I settled on using an organic insecticide called Bti, which kills specifically kills mosquitoes and nothing else. It also does no harm to the surrounding environment.
- The natural question arose: How will I transport the Bti?
  - This is where the thought of using a drone-based system came in. Using a drone to transport the Bti would not only be much quicker and more efficient, but the drone can aid in the effort to spread the Bti conservatively while still exterminating a proportionate number of mosquito larvae.

Now, let’s tackle the main challenge!

I feel that at the present time, it is imperative to first develop a sense for the challenges of my project, and take this in baby steps as to fulfill this vision without taking a rushed approach and moving on. My first course of action is to have a deep learning model, Inception-v3, that has already been trained on a data set I have provided, to test my algorithm that attempts to maximize efficiency and minimize computing power. After doing this, I will design an image classifier that will be catered to my specific classification problem, so as to increase its simplicity while making it more directed in its scope. After having a fully trained and functioning deep learning model, my goal is to have it be uploaded on a drone with autonomous capabilities so that it can first, within a certain range, take pictures, classify the images, and then narrow it down to a location with mosquito larvae, based upon its training. After locking onto their location, a powdered form of Bti, a mosquito insecticide will be dropped effectively killing the mosquito larvae, while keeping everything in the surrounding environment safe.

Basics of Machine Learning

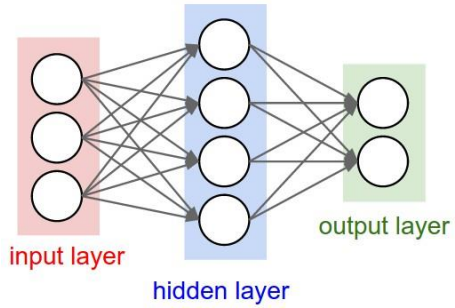
- There are two main types of Machine Learning:
  - Supervised: In this form of Machine Learning, the person creating the neural net knows exactly what the inputs for the network are, as well as what the correct, corresponding output variables should be.
    - Regression Problem: In a regression problem, the output is expected to be some real value that the neural network had been trained to predict.
    - Classification Problem: On the other hand, a classification problem deals with putting the output into a set of specified categories based on the neural networks' inputs.
  - Unsupervised: In this form of Machine Learning, the person who has created the neural net only has knowledge of what is being inputted, however, they do not know what should be the corresponding, correct output variables.

Classification Problem Details

- Gradient Descent: This is a type of algorithm that continuously optimizes a set of parameters as to lower the cost of a given function.
- Cost Functions: Cost functions calculate the cost, or how far from a known value the predicted value, or hypothesis is. The goal is for the cost of a predicted value to be as close to zero in difference from the known value.

Basics of Neural Networks

- Basic Diagram:



  - This is the most common type of neural network as it has fully-connected layers, which means all the nodes in one layer are connected to all the nodes in an adjacent layer. Neural Networks can also be seen as acyclic graphs, as the output of one layer becomes the input of the next one.

- Components of Deep Neural Network: A Neural Network is comprised of three basic components:
  - Input Layer: This layer represents all of the features of an input data set (# of nodes = # of features).
  - Hidden Layer: This layer mathematically implements a weighted sum of its inputs at every node, followed by an activation function like the sigmoid function. In a deeper neural network, the hidden layers are trying to extract different features from the inputs.
  - Output Layer: The output layer is always fully-connected and output "scores" for the desired classes.

- Why Convolutional Neural Networks?: In contrast to a fully-connected layer, a CNN has one node connected to only a few of the total nodes in the adjacent layer. This technique allows for fewer computations by reducing the number of incident edges to the node, without compromising the accuracy in scoring classes. In the case of Deep Neural Nets, the abundance of features requires that these techniques be used for efficient but accurate computing.

About Tensorflow and Inception Model

- Tensorflow is an open source software library from Google for Python and C++ optimized for Machine Learning.
  - Tensorflow helps in performing numerical computations utilizing data flow graphs, which are essentially neural networks.
  - I used Tensorflow 1.0 on Ubuntu 14.04 running on a core i7 CPU with an Nvidia GeForce GTX980 GPU. Tensorflow was configured to work with CuDNN library, developed by Nvidia for parallel computing of neural networks.
- The Inception Model is a special model that has been trained specifically for the ImageNet Large Visual Recognition Challenge. This model has been trained to score 1000 unique classes and can be retrained for a different number of classes.
  - In my case, I retrained this model for four classes of my choice that previously did not exist in Inception.
    - By retraining Inception for these classes, the hidden layers remain untouched, but the number of input features and output scoring and network change.

Retraining Inception for New Classes

- In the scope of this problem, creating an entirely new image classifier for the time being was unnecessary. For a quick prototype, the ImageNet based Deep Neural Network Model, Inception, was employed to classify for a very specific set of categories.
  - Waterbodies: Outputs similarity to waterbody based on set of images given to train off of.
  - “Non-Waterbodies” : Outputs a value that determines similarity to all things other than waterbodies.
  - Mosquitoes: After completing the stage of the classification of a single image to the point that the entire image is a waterbody, the algorithm will output a value based on the similarity it has to mosquitoes (once again, this image will be continuously segmented until it is no longer necessary.)
  - “Non-Mosquitoes” : Works in the same way that the classification for “Non-Waterbodies works.
- The images used to retrain the Inception model for the above categories were gathered from online images, as well as images that were taken by hand, as well as by drone. By using a drone to take pictures to train Inception, the retrained network is rewired to classify images from the view of a drone.
- Since Spring is a time for mosquitoes to breed, it was the optimum time to get to nearby areas to explore stagnant water bodies and collect images to use in the training of the model.

Hierarchical Image Classification Algorithm

- This algorithm's key feature is the way it operates **hierarchically**:
  - The first phase of this algorithm is **reconnaissance**. In reconnaissance, the drone, at a given height H, will survey the geographical features for a given range. Within this given range, an imaginary 2 tile by 2 tile plane will be created. The neural net will then save the tiles with the highest weighted waterbody class, with their respective GPS position and height, H.
  - The next phase of this algorithm is to now perform the downward descent. The drone must reach a height such that the marked and saved tile from height, H, takes up the entire field of vision of the drone. Let us call this new height, h_0. Now, likewise to the "tiling" performed at height H, the same 2 tile by 2 tile will be performed at h_0. In the same fashion as at height H, the tiles with the highest weighted waterbody class will be saved with GPS location, and height.
  - This process of **reconnaissance** and **descent** will continue until the neural network can detect mosquito larvae.
  - This hierarchical approach works well as it has the ability to figure out which tiles are not of importance within a given geographical area, making it much easier to perform reconnaissance after it is done at height H.
  - If the drone camera quality is good, the downward descent can be avoided and instead cropping of the image can be used for the tiles with potential water body. This can make the **reconnaissance** phase limited to single height H.

Hierarchical Image Classification Visual Representation

Mathematics Behind Number of Iterations to Run Algorithm

- Let us say that at a given height *H* a picture with (p × p) pixels is taken by a drone.
  - Let's say that this image's real-world dimensions on the ground are N × N meters ^{2}.
  - Based on the capabilities of the drone's camera we can be certain that if it flies at a height $h < H$ then the image can have the greatest weighted class for mosquitoes. In other words, this height *h* represents the lowest possible height the drone can go, and classify for mosquitoes as the highest weighted class.
  - Now assume that the real-world dimensions of the image taken at height *h*, are n × n meters^2:
    - To calculate the number of iterations or divisions needed to be made in the image, take the ratio of the sides of the two plots of land, *N* and *n*, represented by the images taken by the drone at heights *H* and *h*, respectively.
    - Since we know, every time a division is made in the image, the new side length is half the original, then $\frac{N}{n}$ is equal to some power of 2.
    - If x is equal to the number of iterations, then the following relationship emerges:
      - $x = \log_2\left(\frac{N}{n}\right)$

Other Considerations

- Speed of Computation: The speed at which the drone, or rather the image classifier is able to score an image determines how fast the drone can move over a given plot of land. If it goes too fast, it will be completely overloaded as it attempts to handle taking the picture, moving the drone, and scoring the picture. On the other hand if the drone goes too slow, then efficiency will be lost and the drone will not be able to cover a plot of land in a reasonable amount of time. Since the speed of computation is the one constant, the maximum speed at which the drone can move should be dependent on this speed.
- Capabilities of the Drone:
  - Battery Life: Since the battery of the drone will be controlling both the flight of the drone as well as the classification of the images, it is imperative consider the maximum plot of land the drone can cover with this battery.
  - Maximum Weight: By knowing what the max payload the drone can carry, the amount of battery power being used just by carrying the weight of the drone and other materials, can help determine how far the drone will be able to travel on a single, full charge run.

Future Steps

- Building off of the Hierarchical Image Classification Algorithm, my next course of action is to begin the development of my own unique image classifier that is specifically trained to classify for waterbodies, non-waterbodies, mosquitoes, and non-mosquitoes.
  - This will allow for image classification to take place with less computing sacrificed, as it is know optimized for just four classes, where as the Inception model I'm currently utilizing has been originally trained for many classes, which takes days to train.
- Using this same technique of image recognition, I want to work towards eventually using a Jetson TX1/TK1 by Nvidia as a means of performing classification of images as it is in the air. Non only this, but the camera that feeds data to the Jetson can also be used as a means of adding autonomous capabilities to the drone, so that it can maneuver without any assistance.
- The ultimate goal is to have a fully functional prototype drone that has a pre-trained neural network pre-trained for the classes mentioned above, that is uploaded to a Jetson that is feeding data to the processor in the drone, controlling its motor functions. To accompany this, the prototype will also have a release mechanism for the Bti insecticide.

Acknowledgments

- I greatly appreciate and acknowledge the support of my father, who gave me critiques to my ideas, allowing me to build upon and improve them. My parents provided me an opportunity to build a PC system with GPU that allows massively parallel computing and buy me a drone for data collection.
- In addition, I greatly acknowledge the guidance of Dr. Yong Jae Lee, Assistant Professor in the UC Davis Department of Computer Science for giving me a general pathway and potential challenges I may face.
- I would also like to thank Intel for providing me with a CPU in the past science fair, allowing me to pursue this project.

Sources

"Deep Learning" by Ian Goodfellow and Yoshua Bengio

https://www.tensorflow.org/

http://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/

http://www.fon.hum.uva.nl/praat/manual/Feedforward_neural_networks_1_2__The_classification_pha.html

https://page.mi.fu-berlin.de/rojas/neural/chapter/K7.pdf

http://courses.washington.edu/css490/2012.Winter/lecture_slides/05b_logistic_regression.pdf

http://cs229.stanford.edu/materials.html

http://davidstutz.de/wordpress/wp-content/uploads/2014/07/seminar.pdf

http://neuralnetworksanddeeplearning.com/chap3.html

https://arxiv.org/pdf/1603.05201.pdf

https://www.cs.cmu.edu/~dst/pubs/byte-hiddenlayer-1989.pdf

https://www.epa.gov/mosquitocontrol/bti-mosquito-control