CS230

Employing a Deep CNN to Track Infection Risk for Schistosomiasis from Satellite Imagery

Zac Espinosa
Computer Science
`zespinos@stanford.edu`

Ben Gaiarin
Symbolic Systems
`bgaiarin@stanford.edu`

Michael Vobejda
Computer Science
`mvobejda@stanford.edu`

Abstract

Schistosomiasis is the most common waterborne parasitic disease in Sub-Saharan Africa. Effectively targeting and controlling the freshwater snail populations that host the schistosome parasite can interrupt the disease cycle. Recent advances in high resolution satellite imagery have allowed for us to see where sources of snail habitat are, but analysis of these areas by hand is costly and time intensive. This project is illustrates that a convolutional neural network can be used to effectively measure localized infection risk for schistosomiasis by classifying snail habitat from satellite imagery. We use a dataset of 3,918 satellite images, and rank the presence of two types of vegetation, with each ranking falling in one of nine possible classes. Each class corresponds to the percentage of floating and emergent vegetation present in the satellite image. After testing a variety of architectures, we settled on a branching convolutional network that is trained to generate a heat map that highlights areas of increased infection risk for schistosomiasis.

1 Introduction

In this project we investigate infection risk for the parasitic disease schistosomiasis in communities of endemic areas of Senegal, West Africa. According to the World Health Organization, over 700 million people around the world are at risk of contracting schistosomiasis, with risk of infection being particularly high in sub-Saharan Africa[6]. In order to generate well-targeted efforts to control the spread of schistosomiasis in Senegal, it is important to equip researchers with a low-cost means of determining risk of infection at different community sites.

The parasitic schistosome worm relies on two hosts to complete its life cycle: a human host and a freshwater snail host. Research has shown that measuring the presence of freshwater snail vegetation can provide an indicator for risk of schistosomiasis infection, and that remote sensing techniques (image recognition via satellite imagery) provide an effective route for obtaining those measurements[5]. Currently, researchers have to spend a lot of time tracking the presence of infected freshwaters snails by manually counting snails and using drone imagery to identify snail habitat. Working in collaboration with the lab team of Hopkins Marine Station Professor Giulio De Leo, we have built a convolutional neural network (ConvNet or CNN) that predicts the presence of freshwater snails, and, therefore, indicates risk of infection for schistosomiasis. The input to our network is a collection of satellite images that constitute a given river site in Senegal, West Africa. Our model outputs a heat map of the entire river site, with all images stitched together and selectively tinted to represent localized areas that contain snail habitat and, thus, a heightened infection risk.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

Using high resolution satellite imagery for object detection and image classification has been a growing topic of research in recent years. While many have found success in generalizing standard object detection and classification architectures for satellite or aerial imagery, others have found that for specific tasks additional attention is necessary[7]. Adrian Albert, Jasleen Kaur, and Marta C. Gonzalez use a deep ConvNet and satellite imagery to explore patterns in urban environments at a large scale[1]. Albert et al. compare a VGG16 architecture and a ResNet-50 architecture fed into a 10 class softmax for disctinct land uses. The group found that the VGG16 architecture showed improved accuracy of roughly 80%. Their problem is similar to ours in that it requires a multiclass output of roughly the same size. However, in our problem each class is related to other classes by proximity (e.g. class 4 is closer to 5 than 7). As a result, we choose to define accuracy metrics quite differently.

Similarly, Otavio A. B. Penatti, Keiller Nogueira, and Jefersson A. dos Santos compare two state of the art pre-trained networks, Overfeat and Caffe, in order to differentiate between coffee and non-coffee crop tiles[4]. The group achieved a high accuracy by combining multiple ConvNets. This task is similar to ours in that it involves differentiating landscapes and vegetation types; however, it is slightly simpler in that the task requires binary classification and each input tile shares significantly fewer features than floating and emergent vegetation. Castelluccio et al. use two different proposed architectures, CaffeNet and GoogLeNet[2]. They test these on a variety of datasets, including the Brazilian Coffee Scenes dataset as used by Penatti et al., and find an improved accuracy of 5%.

Hamida et al. explore deep learning for semantic segmentation of remote satellite imagery. Because our project outputs a detailed heat map of floating and emergent vegetation, their work enticed us to strongly consider employing semantic segmentation[3]. However, floating and emergent vegetation typically assume nonuniform and ill-defined boundaries. Furthermore, this group depended on satellite imagery with rich spectral content, and the remote satellite imagery we are provided with lacks this detail. A more in depth discussion of possible future work motivated by Hamida et al. is included in section 6.

Finally, Yi Yang and Shawn Newsam compare different image descriptor techniques such as SIFT descriptors and Gabor texture features[7]. Image descriptors attempt to find salient locations in an image and extract features that are applicable to different images under different conditions. This technique of classifying remote sensed imagery does not involve deep learning. It is worth noting that there exist alternative techniques to solving this problem and such techniques set the benchmark for current performance.

3 Dataset and Features

Our data set is comprised of 48 satellite images (courtesy of Planet Labs) of endemic water site areas of the Senegal River Basin. Each satellite image (.tif file) has a resolution of 3 meters, is between 80MB and 180MB, and has dimensions 4000 x 2000 x 4 pixels. In order to generate our training set we break each of these satellite images into 150x150x4 sub-images (when referencing satellite images in the remainder of this paper, we will be referring to these 150x150 sub-images unless explicitly stated otherwise). We employ a 9-class softmax to label each image with two rankings (1-9) that indicate the prevalence of two different types of vegetation that provide habitat for schistosome-carrying snails: floating ('in the water') and emergent ('on the shore') vegetation. A 1 corresponds to no vegetation, a 2 corresponds to slight vegetation, and so one until 9. Classified image examples can be seen below in Figure 1, Figure 2, and Figure 3. Figure 2, for example, is labeled 1-9, implying a complete coverage of the shoreline in emergent vegetation and no presence of floating vegetation in the image.

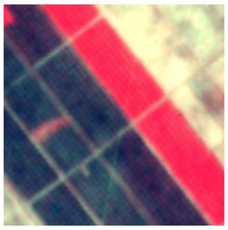


Figure 1: Label 1-1

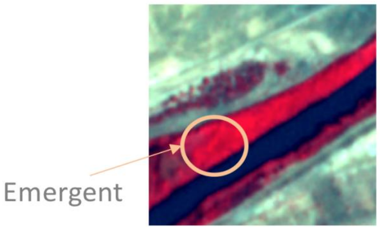


Figure 2: Label 1-9

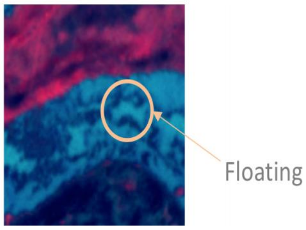


Figure 3: Label 8-9

We manually classified approximately 4,000 sub-images and use data augmentation to expand our data set to over 10,000 images. We noticed that our data set was skewed, with approximately 80% of images being classified 1-1. As a result we run data augmentation only on images with a classification of greater than 1 for either floating or emergent vegetation. This brings our ratio to approximately 40% 1 labels and 60% non-ones. Furthermore, after each epoch we normalize the data by dividing each image by 127.5 (half of max pixel value). A discussion of the effects of normalization and data augmentation are in section 5. Finally, because we have a fairly small data set we use a training and dev set of 95% and 5% respectively.

4 Methods

The model begins with three convolutional layers followed by a maxpool layer, a convolutional layer and a max pool layer. The first of these convolutional layers uses a relu activation function and the next three layers use a tanh activation function (Please reference Figure 4 for a graphical representation of our final model). Moreover, both max pool layers use a 4x4 filter and stride of 2. The network follows a depth of [16, 32, 64, 64] such that an image flowing through this network has an initial depth of 4 and final depth of 64. The model then branches into two networks, the first committed to floating and the second committed to emergent. Floating and emergent vegetation share similar features and the initial network is used to learn shared features. The branched networks are used to learn nuances between the two types of vegetation. Each branch contains an additional two convolutional layers both with relu activation functions and of depth 128. These layers are followed by two fully connected layers, the latter of which uses a softmax of class 9. All weights are initialized using xavier initialization. This model follows the standard form where images decrease in vertical and horizontal dimension but increase in depth throughout the network. We use a composite cross entropy loss function:

$$L_f(y_f, \hat{y_f}) = y_f log(\hat{y_f}) * w \tag{1}$$

$$L_e(y_e, \hat{y_e}) = y_e log(\hat{y_e}) * w \tag{2}$$

$$J(W, b) = -\sum_{i=1}^{M} L(y_{fi}, \hat{y_{fi}}) + L(y_{ei}, \hat{y_{ei}}) \tag{3}$$

$L_f(y_f, \hat{y_f})$ is the loss function associated with floating vegetation and $L_e(y_e, \hat{y_e})$ is the loss function associated with emergent vegetation. Note that w is a weight associated with the class that this loss is being computed for. By creating a composite loss function we train our network to detect both floating and emergent vegetation. Crucially, the cross entropy loss function can be used for multi-class problems such as ours. It helps find the difference between prediction and true value and attempts to narrow/converge predictions to true values. Before settling on this architecture we experimented with multiple models, with the most competitive being a model of the exact same architecture, yet excluding the last two convolutional layers after branch. This first version of the architecture for the model branched directly into two fully connected layers and generated marginally worse results, but significantly improved runtime and, so, we strongly considered employing this architecture in our final model.

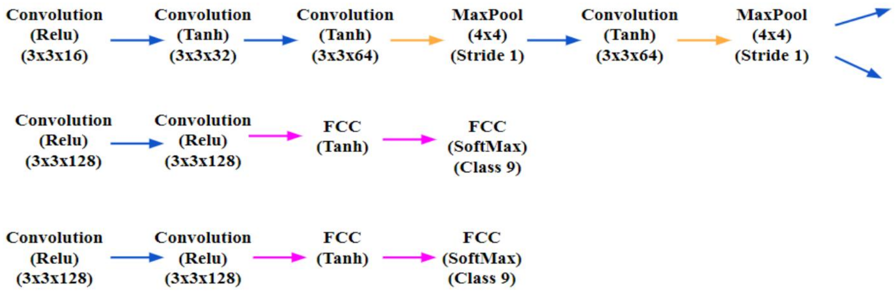

Convolution (Relu) (3x3x16)
Convolution (Tanh) (3x3x32)
Convolution (Tanh) (3x3x64)
MaxPool (4x4) (Stride 1)
Convolution (Tanh) (3x3x64)
MaxPool (4x4) (Stride 1)
Convolution (Relu) (3x3x128)
Convolution (Relu) (3x3x128)
FCC (Tanh)
FCC (SoftMax) (Class 9)
Convolution (Relu) (3x3x128)
Convolution (Relu) (3x3x128)
FCC (Tanh)
FCC (SoftMax) (Class 9)

Figure 4: Final Architecture

5 Experiments/Results/Discussion

A link to the final results of our experimentation, including all code we wrote for the various components of our task, can be found here: `https://github.com/zespinosa/cs230_project`.

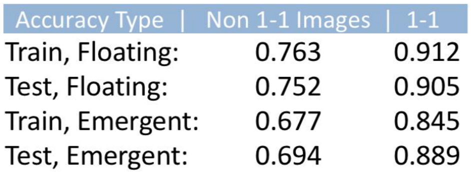
Our primary performance metric for our model is accuracy. Similar to our cost function, accuracy is a composite function; for images that have a true label greater than 1 we take the difference between the predicted class and the true class. If this difference is less than or equal to 3 than the prediction is accurate, and otherwise the prediction is inaccurate. For images that have a true label of 1 we take exact equivalence for our accuracy. After discussing the problem with Professor Giulio de Leo'team, we agreed that is it worse to incorrectly detect the presence/absence of vegetation than to incorrectly detect the quantity of vegetation. As a result, our accuracy demands exactness for labels of 1 and lenience for labels otherwise. This lenience also aligns with the inconsistencies/subjectiveness of manual data labeling as discussed in our future works section (section 6).

In order to make our cost align with accuracy for this project we experimented with a variety of cost weightings for each class. We decided that class 1 should be weighted more heavily than others, and after additional testing settled on a class 1 weight value of 1.5.

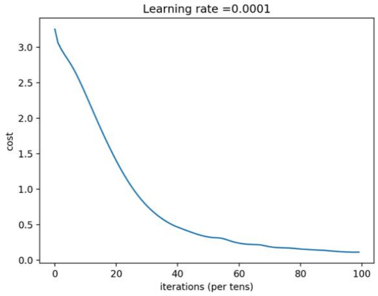
Our final learning rate for gradient descent is .0001. We tested learning rates of .01, .09, .001, and .0001, and found that .0001 worked the best. This aligns with what we expect; since the variance of our training data is extremely high, taking smaller steps reduces jumping, smooths our cost function, and allows us to learn much faster.

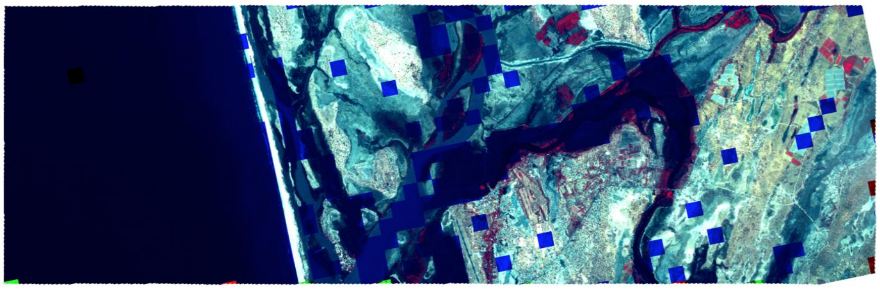
After experimentation with mini-batch sizes of 1, 16, 32, and 64 we settled on a mini-batch size of 16. This mini-batch size is fairly small and allows us to take many steps per epoch. This counterbalances an extremely low learning rate.

Moreover, in order to determine our predicted label from the softmax layer we experimented with two approaches: taking the argmax and taking the mean over classes. We found that the argmax best aligned with our definition of accuracy as stated above. This is because our model outputs a vector of length 9 with values that are trained to minimized our cost function. As described above, our model (predictably) outputs vectors that have negative values in the indices that are farther away from the index it thinks is most likely the correct label for each type of vegetation. When the difference between the output vector and the one-hot label vector for a given training example is calculated, the negative values offset inaccuracy from the index containing the 1 in the label. To correctly glean the predicted label from the output vector of our model, we use an argmax function. This returns the index containing the highest value from the output vector, which is the label (1-9) that our model thinks is the most likely rating for the given training example.
The following accuracies, cost function, and heat map represent the results of our experimentation:



Accuracy Type	Non 1-1 Images	1-1
Train, Floating:	0.763	0.912
Test, Floating:	0.752	0.905
Train, Emergent:	0.677	0.845
Test, Emergent:	0.694	0.889





(from top left to bottom) Figure 5: Final Accuracies, Figure 6: Cost Function, Figure 7: Heat Map

The cost function in Fig. 6 is what we would expect to see: an exponential decay in cost that suggests adequate learning. Based on the accuracies presented in Fig. 5, the model does well to correctly label 1-1 images with no vegetation, but fails to classify non 1-1 images with sufficient confidence. However, Fig. 7, which represents the final output of our model and the metric we care most for, suggests otherwise. The heat map in Fig. 7–with emergent tinted in blue, floating in red, and grids with both vegetation types in green–suggests adequate precision for emergent vegetation classification; we see most areas with emergent vegetation in Fig. 7 correctly tinted blue. On the other hand, Fig. 7 also shows very few red and no green squares–this is partially due to the fact that the land area has little floating vegetation to begin with, but it nonetheless suggests that more floating training data is necessary for the model to adequately learn floating vegetation. Finally, we find numerous false positives in Fig. 7 where grids without vegetation have been classified as having emergent vegetation (tinted blue); we can infer that a larger training set with a more rigorously consistent classification scheme is necessary to reduce this false positive count.

6 Conclusion/Future Work

This project is a proof of concept, illustrating that satellite imagery can be fed through a convolutional neural network to effectively measure infection risk of schistosomiasis for river side communities in endemic areas. We found that a CNN can achieve moderate accuracy in this task. We believe that a branching CNN, as described in Methods, outperforms other models for a number of reasons. Firstly, by passing both the emergent training set and floating training set through the same initial network, we are able to capitalize on shared low level features and improve training time. By then branching our model, we are able to train two distinct networks to precisely distinguish between floating and emergent. Furthermore, by using a cross entropy loss function and nine-class softmax output layer, we are able to more accurately measure not only the presence of floating and/or emergent vegetation, but also the quantity in each satellite image.

While creating this model we compiled a set of key design decisions that will be considered by Giulio de Leo's lab team in their final implementation of the model. First, we found that there must be an agreed upon methodology for classifying images and that the individuals classifying must be appropriately trained. Despite our best attempts to create a standard methodology for classifying images, our data set had an extremely large variance due to our lack of professional training. Reducing this variance and standardizing labeling would greatly improve the performance of any network. Employing satellite imagery of a higher resolution could also help reduce variance. Furthermore, in addition to the satellite imagery provided to us, the lab team has access to high quality drone imagery of the Senegalese sites they conduct field work in. This drone imagery is costly and difficult to obtain, which is why the final implementation of this model will use exclusively satellite imagery to generate a heat map, as we have done. However, the drone imagery is still valuable, for it can be cross referenced in the manual labeling process to obtain more accurate and consistent labeling.

Our heat maps, while they suggest adequate recall for this first draft of the model, show us that our model generates too many false positives, and that the precision for the model requires improvement. We found numerous tinted grids in our heat map that did not contain water, yet were not classified with a rank of 1-1 (no vegetation). Given more time, we would have tested a two step architecture. The first step would be a simple CNN for binary classification. This CNN would determine whether a satellite image has floating vegetation, emergent vegetation, both, or no vegetation. The second step would be nearly identical to our current model, expect it would have an 8 class softmax, where the first class implies the slight presence of vegetation. Employing such a two-step architecture would enable our model to specialize in the two actions involved with our task: detecting presence of vegetation, and quantifying amount of vegetation.

Ultimately, we hope that this preliminary model and report will guide professor Giulio de Leo's lab team in their final implementation. We trust that future implementations of this model will offer an effective, low-cost, accurate means of measuring the extent to which communities are at risk of coming into contact with schistosome-contaminated snail populations.

7 Contributions

- Zac Espinosa: Creator and manager of GitHub repository, author of model.py milestone and contributor to final design, planning and creation of model.py. Author of importTiff module. Classified approximately 1,000 images. Contributed to hyperparameter exploration. Implemented random minibatching.

- Ben Gaiarin: Communications with Giulio de Leo's lab, background research, author of project proposal, milestone write ups, and poster. Responsible for data collection and initial exploration. Classified approximately 1000 images. Author of data augmentation, implemented weighted cross entropy loss function, and contributor to final design, planning and creation of model.py.
- Michael Vobejda: Author of image classifier, allowing team to quickly label 3000 images. Classified approximately 1000 images. Author of createheatmap.py: module used to applying tinting and build heat map. Adapted model.py to have end to end work flow, implemented data normalization, contributed to final design, planning and creation of model.py.

References

[1] Albert, Adrian, Jasleen Kaur, and Marta Gonzalez. "Using Convolutional Networks and Satellite Imagery to Identify Patterns in Urban Environments at a Large Scale." (2017): n. pag. Web.

[2] Castelluccio, Marco et al. "Land Use Classification in Remote Sensing Images by Convolutional Neural Networks." (2015): n. pag. Web.

[3] Hamida, A Ben et al. "Deep Learning for Semantic Segmentation of Remote Sensing Images with Rich Spectral Content." n. pag. Web.

[4] Penatti, A B, Keiller Nogueira, and Jefersson A Santos. "Do Deep Features Generalize from Everyday Objects to Remote Sensing and Aerial Scenes Domains?" (2015): 44–51. Web.

[5] Walz, Yvonne et al. "Modeling and Validation of Environmental Suitability for Schistosomiasis Transmission Using Remote Sensing." *PLoS Neglected Tropical Diseases* 9.11 (2015): n. pag. *PLoS Neglected Tropical Diseases*. Web.

[6] World Health Organization. "WHO Schistosomiasis : Countries x Indicators." *Schistosomiasis*. N.p., 2010. *Schistosomiasis*. Web.

[7] Yang, Yi, and Shawn Newsam. "Comparing Sift Descriptors and Gabor Texture Features for Classification of Remote Sensed Imagery." *IEEE International Conference on* (2008): 1852–1855. *IEEE International Conference on*. Web.

Link to Code Repository

`https://github.com/zespinosa/cs230_project`