# Final Project: *Hole and Tunnel Detection Using Convolutional Neural Networks*

Munir, Saima Binte
*FEAS, Department of ECBE*
*Ryerson University*
Toronto, CA
saimabinte.munir@ryerson.ca

Rashidi, Tabish
*FEAS, Department of ECBE*
*Ryerson University*
Toronto, CA
tabish.rashidi@ryerson.ca

Martinovic, Daniel
*FEAS, Department of ECBE*
*Ryerson University*
Toronto, CA
daniel.martinovic@ryerson.ca

***Abstract*— Convolutional Neural Networks are a powerful way of processing images to predict their classification. With several possible applications of this technology, a really important problem it can solve is by distinguishing between holes, tunnels which may have implications in search and rescue research. This experiment constructs a CNN to perform this application and attains around 70% accuracy and may become even more accurate after identifying and correcting the relevant parameters.**

***The source code of the project can be found at***

**https://github.com/saima1999/Hole_Detection**

***The dataset can be found at***

**https://drive.google.com/drive/folders/1-sfDBxBmfRrNW0Uc3k3xhUbz6XxXVrOf?usp=sharing**

## I. Introduction

During 2001-2010, an average of 700 natural disasters occurred every year, affecting a total population of 270 million [1]. Natural disasters are not only devastating to people, but also city infrastructure. The destruction of such structures can cause the entrapment of those within. Studies have shown that these people are indeed surviving while trapped, from twelve hours to as long as 6 days after the incident [2]. However, this is still not a safe position to be in as buildings can still collapse much after the initial damage. The key for search and rescue teams is to reduce the time spent finding trapped individuals and at the same time, reducing risk to themselves. One solution to this problem is unmanned autonomous robotic systems that are able to detect possible entrances, like a tunnel or holes where people may be trapped. Identifying key places like this autonomously, allows search teams to respond faster.

This project attempts to solve this problem of labelling holes, tunnels and plain objects through a predictive classification system using Convolutional Neural Networks (CNN). The CNN model contains 4 convolution layers, with each layer performing a convolution operation, followed by the activation function being applied and a max pooling operation. After these convolution layers, a flatten function and a linear function are applied. The model is then trained and key metrics like training losses, validation losses, and validation accuracy are calculated to measure the performance.

### *Past Work*

Kong et al. has done similar work in the detection of holes in search and rescue sites [3]. In fact, it is one of the first papers in its field of using holes to identify access points for search and rescue applications. It defines the concept of “holes" as *a potential for access into a collapsed structure*. The paper also implements a method for hole detection using a RGB-D camera, to detect and isolate the hole feature from a UAV image. The algorithm is discussed in-depth in the *Comparison to Other Approaches* section.

The other closely related paper is *Mind the gap: detection and travers-ability analysis of terrain gaps using LIDAR for safe robot navigation* by Sinha and Papadakis [4]. In this paper, the authors address the issue of gaps and what are known as *negative obstacles*. They propose a method to detect these gaps and were able to successfully implement this on a contemporary robot that was able to traverse and/or avoid these gaps.

## II. Technical Explanation

Machine learning concepts of classification as well as CNNs and loss functions are crucial to implementing this computer vision solution of identifying holes, tunnels, and plain objects.

Classification is a supervised learning technique of using given data points (i.e. input variables, X) to predict the discrete output variable, y and label it as a class [5]. This is done through approximating a mapping function for the input variables to the output variables. Multi-class classification is used in this

project to label the 3 classes of holes, tunnels and plain objects, using predictive modeling.

To perform this, 2 major parts of the algorithm are required, which are the feature learning and classification portions. Feature learning has a CNN to perform convolution, apply the activation function and perform pooling. Classification includes using a flatten function, having fully connected layers and applying the SoftMax function.

For feature learning, a CNN is a type of artificial neural network, which is a network of artificial neurons. A neuron is a basic structural building block and allows for the forward propagation of information through it. It functions by taking inputs (i.e. an array of pixel values from images) and performing a dot product with weights, adding a bias, and applying a non-linearity/ activation function [6].

Let y be the output, g be the non-linear activation function, xi be the linear combination of inputs, $w_i$ be the linear combination of weights, and wo be the bias. Then, the output can be calculated through the equation

$$\hat{y} = g\left[\left(\sum_{i=1}^{m} x_i w_i\right) + w_o\right]$$

This can also be expressed as

$$\hat{y} = g(w_o + X^T \cdot W)$$

where

$$X = \left[x_i, \ldots, x_m\right]$$

$$W = \left[w_i, \ldots, w_m\right]$$

These neurons are assembled into different layers that have different transformation effects on the input. The model learns through adjusting the weights of neurons.

The convolutional network architecture was chosen as it performs better for image processing purposes. In the CNN, patches of input, which is an array of image pixel values, would be connected to neurons in the hidden layers. A sliding window is used to define these connections. Through each progressive convolution layer, the model learns lower level features to higher level features with the convolution operation. Convolution uses filters to perform an element-wise multiply, and add the outputs to generate feature maps. Thus, each neuron (p, q) in each hidden layer takes inputs from the patch, computes the weighted sum using a 3x3 filter, applies the bias using the following equation

$$\sum_{i=1}^{3}\sum_{j=1}^{3} w_{i,j}\, x_{i+p,j+q} + w_o$$

After every convolution operation, the ReLU activation function was chosen for introducing non-linearities to approximate arbitrary complex functions and to replace all negative pixel values by zero. Next, a max pool 2D function was used to down sample for reducing dimensionality and preserving spatial invariance.

For this project, a CNN with 4 layers was constructed. Each layer performs a convolution, then gets activated through the ReLU function, with this the undergoing max pooling operation of a 2x2 sliding window.

During the classification portion, the flatten function was applied to flatten a contiguous range of dimensions into tensors and the linear function was applied to perform a linear transformation. The SoftMax function expressed as

$$softmax(y_i) = \frac{e^{yi}}{\sum_j e^{yj}}$$

expresses the output label as a probability.

For this neural network to evaluate itself, a loss function was used to quantify the cost of the network making wrong predictions. Cross entropy loss was calculated as our model as our model outputs a probability between 0 and 1.

Let W be the linear combination of weights, $y^{(i)}$ be the actual label and $f(y^{(i)}; W)$ be the predicted label, then the equation of the cross-entropy loss is

$$J(W) = \frac{1}{n}\sum_{i=1}^{n} y^{(i)} log(f(y^{(i)}; W)) + (1 - y^{(i)}) log(1 - f(y^{(i)}; W))$$

## III. Experiments

In this section of the report, the experimental data is presented and analyzed. There are 273 total images in the dataset, of which 168 are holes, 16 are plain objects and 87 are tunnels.

While attempting to run the algorithm on the dataset, there was a clear issue with different image sizes resulting in unequal tensor sizes in the TensorFlow model. To mitigate this error, the images were resized to a smaller size of 60 pixels x 80 pixels, to allow for rapid testing and training. Although there is some data loss in the images, the major features of the images are still detectable.

When segregating the data into different sections for training and testing the model, because the dataset is relatively small, a 90-10 split was used in which 90% of the data was used as the training set and 10% was used as the validation set.

When considering the accuracy of the results, it is clear that there is an effect from the training data. There is an over-representation of holes, and a significant under-representation of plain objects. This will impact the model as it can accurately predict a hole with good confidence but skew the predictions for the other two object types.

The experiments were conducted on a system with 16GB RAM, processor: 3.6 GHz, 6 cores, on a UNIX based operating system. The training time for the model with the given dataset is about 1 minute, when utilizing all cores.

*Analysis*

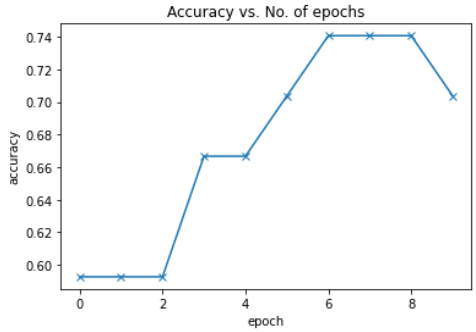


*Figure 1. Validation Accuracy Over Epochs*

In *Figure 1*, we can see the Accuracy vs. Number of Epochs. The graph shows an upward trend in the accuracy of the model as the number of epochs goes up, this is expected for machine learning models, showing a convergent pattern. Since there is a slight drop in accuracy near the end, that could be considered a false local minimum, which indicates that our learning rate of 0.001 might be slightly too small. In addition, the actual accuracy of the model hovers near 70% by the end of the epochs, which means the model is on the right track, but more adjustments must be made to the parameters.

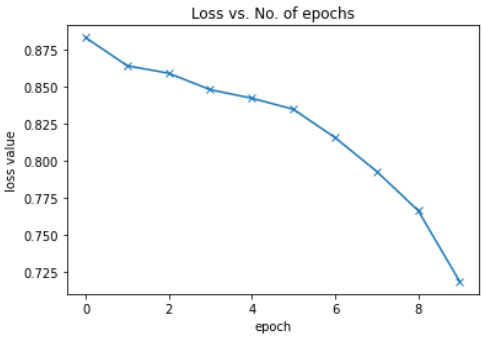


*Figure 2. Loss Over Epochs*

When comparing the training loss with the validation loss, the validation loss is slightly larger than the training loss. This means that the model is slightly overfitting to the data it is being trained on as it suffers a greater loss when being trained on the new data of the validation dataset, compared to the training dataset.

To further tweak the model, perhaps the learning rate should be adjusted, more convolution layers could be added, and the parameters for the convolution could be changed. For instance, the dimensions for the number of channels in the input image and number of channels produced in the output could be changed. The dilation, which is the spacing between kernel elements, and the stride value of the convolution could be adjusted to produce better results.

The following data is procured from the validation set. We have randomly selected images from holes, tunnels and plain objects.

*Prediction of Holes*

A large part of the dataset consisted of mostly hole images, it is evident that the model is able to accurately identify the holes much better than plain objects and holes.

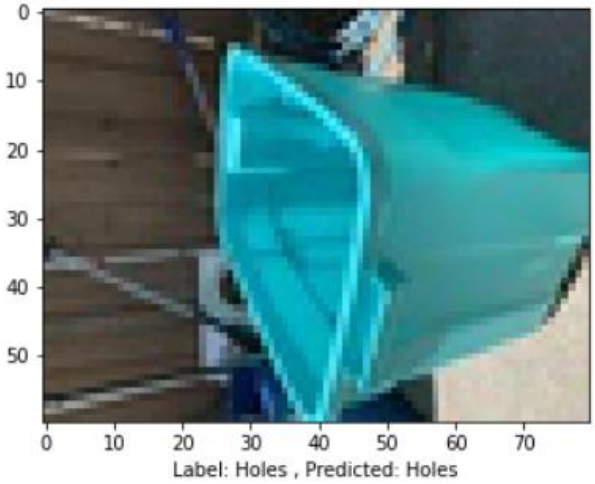


*Figure 3. Correct Prediction of Holes*

Randomly selecting from the validation model, the model is able to repeatedly identify holes quite well.

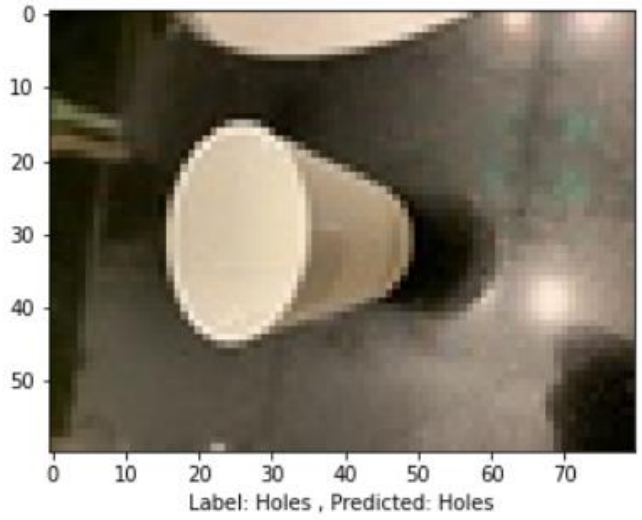


*Figure 4. Correct Prediction of Holes*

*Prediction of Tunnels*

Tunnels are the next largest part of the dataset, and although the model can predict tunnels, it can mislabel images completely. Although *Figure 5* shows the model predicting the tunnel well, the very next randomly generated image in *Figure 6* showed a tunnel but the model predicted it as a hole. Although this is incorrect, a potential cause for this error would be the cup having a shadow which the model would assume is an opening at the end of it, there is some darkness at the end of it, which may have caused the model to assume this as a hole.

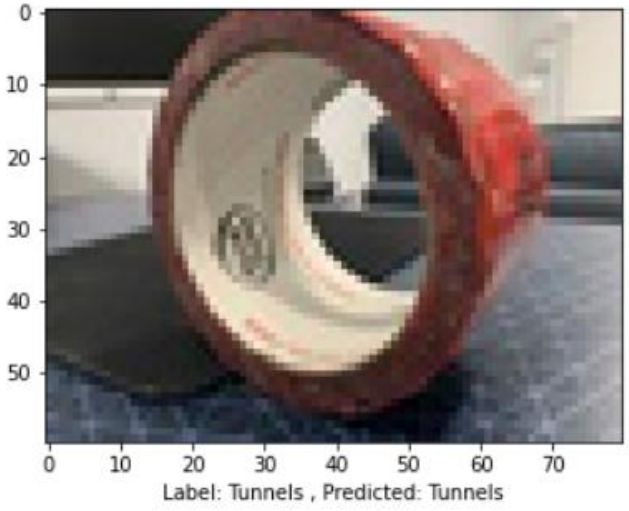


*Figure 5. Correct Prediction of Tunnel*

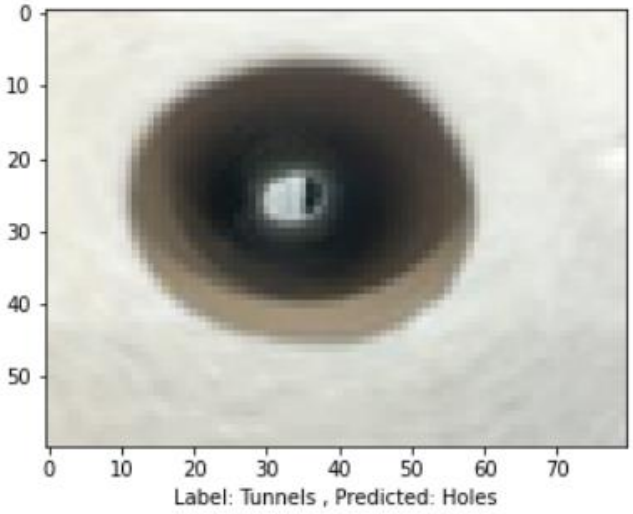


*Figure 6. Incorrect Prediction of Tunnel*

*Prediction of Plain Objects*

Plain objects were the smallest part of the dataset. Hence, it follows that the predictions formed in this part of the dataset are the most inaccurate. *Figure 8* shows that the model correctly predicts the birdhouse as a plain object. However, in *Figure 7*, it inaccurately predicted that the lotion bottle is a tunnel. The model is unprepared to predict positive objects and will need significant training in order to receive a satisfactory accuracy.

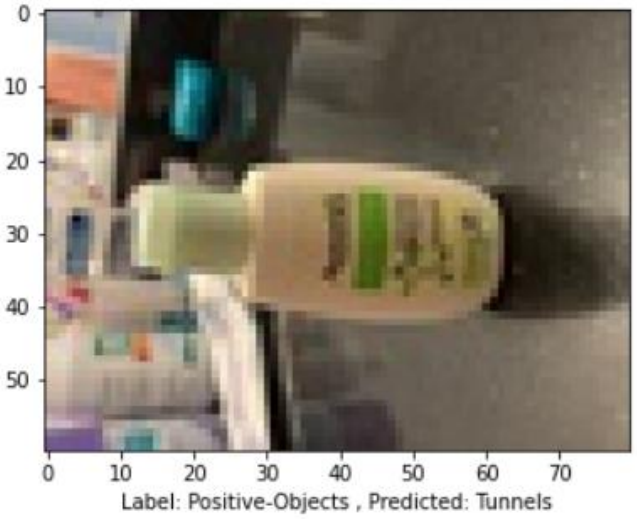


*Figure 7. Incorrect Prediction of Plain Object*


*Figure 8. Correct Prediction of Plain Object*

*Comparison with Other Approaches*

This paper focuses on a ternary approach to classifying different kinds of objects. Other papers have taken different approaches to detect holes and important features in images. The most notable and closely related paper is the first paper mentioned in the Introduction of this paper *What is a Hole? Discovering Access Holes in Disaster Rubble with Functional and Photometric Attributes* by Kong et al. [3]. It outlines a system to detect holes as a feature in images, utilizing more classical computer vision techniques as opposed to a machine learning approach. It first requires the use of a RGB-D sensor to give relevant depth information. Using this information, the image can then be broken up into *super pixels*, a set of pixels that correspond to a similar property in an image. After obtaining multiple super pixels, the algorithm compares the super pixels for its neighbors assigning a weight to indicate the likelihood of a hole. When all super pixels have been evaluated and their features detected, the super pixel with the most weight is detected.

This approach focuses on a feature-based approach to detecting holes in an image. By comparing the pixels in the image, not only is it able to identify a hole but given multiple holes, it is able to correctly identify the largest one.

In contrast, the approach used by this paper focuses on allowing the convolution neural network to extract the significant features of the image where is Kong et al. develop a more robust detection method for holes. Furthermore, Kong et al. use an RGB-D sensor to utilize depth information, which is significantly important to hole recognition, whereas the model presented in this paper uses 2D images with no information on depth.

A more suitable approach perhaps would have been to add two extra layers for *Feature Detection:* one layer to get numerical features of the image, like contrast and brightness, and the second to select the features that correlate strongest to the desired output using a *Correlation Heat Map*.

## IV. Conclusion

In summary, identifying holes, tunnels and plan objects are an important field of interest due to the significant effect of search and rescue applications and autonomous robots used for surveying damage sights. The performance of the CNN was measured to have tested at 70% +/-5% accuracy between the 3 labels, indicating that the model can be further improved. Furthermore, the validation loss of the model is slightly higher than the training loss, proving that this model is slightly overfitting to the training data.

Overall this project has given our team a much better understanding of machine learning and given us the opportunity of implementing a real-world solution using CNNs.

*Project Member Contributions*

*Tabish* - Developed code for core ML model (helper), edited dataset images, wrote report (Experiment, Analysis, Comparison with Other Approaches)

*Saima* - Developed code for core ML model (main), wrote report (Technical Explanations, Experiment, Analysis, Conclusion)

*Daniel* - Developed code for core ML model, (helper), wrote report (Introduction, Past Work, Technical Explanations)

## References

[1] World Health Organization. (2017). Emergency response framework. World Health Organization. http://www. who. int/hac/about/erf/en/. Published.
[2] Macintyre, A. G., Barbera, J. A., & Smith, E. R. (2006). Surviving collapsed structure entrapment after earthquakes: a “time-to-rescue” analysis. Prehospital and disaster medicine, 21(1), 4-17.
[3] Kong, C., Ferworn, A., Coleshill, E., Tran, J., & Derpanis, K. G. (2016). What is a Hole? Discovering Access Holes in Disaster Rubble with Functional and Photometric Attributes. *Journal of Field Robotics*, *33*(6), 825-836.
[4] Sinha, A., & Papadakis, P. (2013). Mind the gap: Detection and traversability analysis of terrain gaps using LIDAR for safe robot navigation. Robotica, 31(7), 1085-1101. doi:10.1017/S0263574713000349
[5] Asiri, S. (2018, June 11). *Machine learning classifiers*. Medium. Retrieved December 19, 2021, from https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623
[6] Alexander Amini. (2020). *Mit 6.S191 (2020): Convolutional Neural Networks*. *YouTube*. Retrieved December 19, 2021, from https://www.youtube.com/watch?v=iaSUYvmCekI&t=1041s.
[7] Jovian. (n.d.). Retrieved December 19, 2021, from https://jovian.ai/aakashns/05-cifar10-cnn