A Convolution Neural Network for Plant Species Identification

Shiang Jin, Chin
Khoury College of Computer Science
Northeastern University
Seattle, WA
chin.shi@northeastern.edu

***Abstract*. In this paper, I describe the training of machine learning models based on convolution neural network architecture for the image classification task of identifying plant species based on the leaves. Throughout the paper, I document the various experiments with different model architectures and their corresponding results. The best-performing model uses a pre-trained ResNet-50 model fine-tuned to classify plant species based on leaves. The model performs better and computes faster on the Leafsnap dataset compared to the recognition system developed before.**

Keywords—convolution neural network, ResNet-50, Leafsnap database

I. Introduction

In this work, I describe the experiments carried out to train a machine learning model based on a Convolution Neural Network to classify plant species based on their leaves. The research idea was inspired by the Greenstand Project[1]. But the datasets used, and the previous works referenced were from the Leafsnap dataset and system [2].

The Leafsnap system utilized a four-step process (see section II.B) where good segmentation to separate the leaf from the background is crucial for the system [2]. The classification is based on computing the histograms of curvature over scale (HoCS) as the feature vectors and comparing them with the precomputed feature of a labeled dataset. The system achieved an accuracy score of 96.8% for the target within the top 5 matches with a computation time of about 5 seconds [2]. While the accuracy score and computation time are good enough for practical daily usage, the question arises can we achieve a higher accuracy score, using a simpler process (without the need for segmentation), and using less computation time?

The convolutional neural network (CNN) is the most famous and commonly employed algorithm in the deep learning field used for image classification. One of the main benefits of CNN is that it automatically identifies the relevant features and the filters required to extract the features without any human supervision [3]. Thus, compared to the system used by Leafsnap, a CNN-based model has the potential to optimize the four-step process together and achieve a better accuracy score. The other potential advantage of CNN based model is it can offer a fixed computation time for the classification once the model is trained, whereas for the Leafsnap system, its computation time will increase with increasing amount of data in the labeled database.

The remaining parts of this report are organized as follows: first I gave a brief overview of the dataset and the related work relevant to this paper. Next, I presented a method to build and train three different CNN architectures. This is followed by experiments and results on the baseline model using different subsets of the data. As the baseline model was unable to achieve the accuracy score desired based on the full dataset, I presented the results of experiments for going wider (using a larger input image for the model) and going deeper (using more color channels for the model or more convolution stacks). Next, I presented the results based on a pre-trained ResNet-50 model fine-tuned for the plant species classification tasks, before wrapping up with the discussion and conclusion section.

II. The Data Set & Related Works

A. The Leafsnap Dataset

The Leafsnap dataset is available at Kaggle (https://www.kaggle.com/datasets/xhlulu/leafsnap-dataset) and was created by computer scientists from Columbia University and the University of Maryland, and botanists from the Smithsonian Institution in Washington, DC [1]. It covers all 185 tree species found in the Northeastern United States.

The images of leaves are taken from two different sources, high-quality images taken in the "lab environment" for pressed leaves from the Smithsonian collection (total of 23915 images), and lower-quality images taken in the "field environment" using mobile devices in the outdoor setting (total of 5192 images).

The images came with two subsets, the first subset is the images in the original color of RGB scale, and the second subset is the images in grayscale after the segmentation step.

B. Related Work I (Leafsnap)

The recognition system built by the researcher for the Leafsnap dataset consists of the following steps [2]:

Classification step to validate if the image is a leaf for further processing, using a binary classifier based on gist features and Support Vector Machine (SVM)

Segmentation step to obtain a binary image separating the leaf from the background.

Extracting step to compute the feature vector representing the shape of the leaf based on histograms of curvature over multiple scales (HoCS). This HoCS is scale-invariant, rotation-invariant, and translation-invariant [2].

Comparison step to identify the tree species based on the nearest neighbors found in the labeled database using histogram intersection as the distance metric.

The system achieved an accuracy score of 96.8% for the target to be within the top 5 results returned.

C. Related Work II (Searching the World's Herbaria)

Searching the World's Herbaria is a system for the visual identification of plant species used before Leafsnap was developed. The system used Inner Distance Shape Context (IDSC) as the main feature vector to classify the plant species

based on the leaves [5]. IDSC tries to represent the shape via histograms of distances and angles from sample points on the contour to all other points, along a path inside the leaf [4].

This approach achieved a top 5 accuracy score of 85.1% on the Leafsnap dataset [2]. While it may be acceptable for daily usage, the quest for higher accuracy led to the development of the Leafsnap recognition system. The Leafsnap has achieved good accuracy scores whereas searching for improved segmentation algorithms and classification models manually has become more difficult. Hence the focus is shifted to let the machine learning model learn itself how best to segment and classify the images using the CNN architecture.

D. Related Work III (Resnet)

For convolution neural networks used for the image classification task, the "levels" of features are known to be enriched by the number of stacked layers (depth) [6] and the network depth has shown to be of crucial importance to the result [7]. Hence if the baseline model is unable to achieve the accuracy score desired, the next step is to try existing network architecture that allows us to go very deep.

ResNet is one of the popular deep network architectures for image classification tasks that allow us to go deeper (by building more convolution layers) while avoiding the effects of overfitting or degradation [6]. It also has lower computation requirements compared to another network of the same depth (for example VGG) while achieving a better accuracy score [6].

III. METHODS

A. The Baseline Model (2CNN)

The architecture of the baseline model is shown in Figure 1. It consists of two convolution stacks followed by a fully connected stack. All model components are derived from the torch nn.Module class.

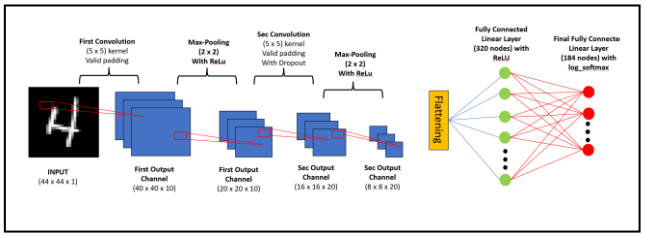
The first convolution stack is built from a convolution layer converting one channel (grayscale image) into 10 channels with 5 x 5 filters and valid padding, followed by a max pooling layer with a 2 x 2 window and a ReLU function at the end.

The second convolution stack is built from a convolution layer converting the ten channels output from the first convolution stack into twenty channels with 5x5 filters and valid padding. Thereafter a dropout layer with a 0.5 dropout rate is applied, before feeding the input into a max pooling layer with a 2 x 2 window with a ReLU function at the end.

Output from the second convolution stack is fed into the fully connected stack for final classification. The output is first flattened into one-dimensional data and passed into the first fully connected linear layer with 320 hidden nodes and the ReLU function applied. The data is then passed to the final fully connected linear layer with 185 nodes (equal to the number of species present in the dataset), with the log_softmax function applied to the output.

The input size for the image is set to be 44 x 44. This will produce enough output nodes (of 1280) at the end of convolution stacks before scaling down to the number of hidden nodes.

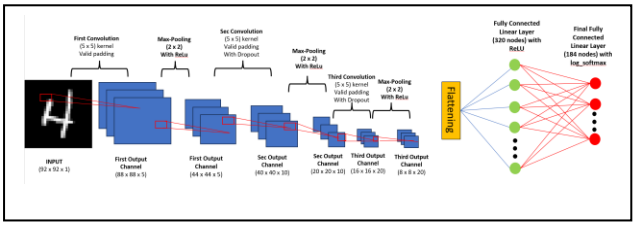
Figure 1: Architecture of baseline model



B. A Deeper Model (3CNN)

A deeper model is built to add an additional convolution stack (filters with max pooling layer and ReLU function) to the baseline model. To ensure the number of output nodes from the convolution layers is equal to the number of output nodes from the baseline model, the number of output channels was adjusted to 5 for the first CNN stack, 10 for the second CNN stack, and 20 for the third CNN stack (See Figure 7). Due to the addition of a third CNN stack, the input image size for the model was also enlarged to 92 x 92 to achieve the same output size of 8 x 8 as the 2CNN model for comparison on the effect of more depth.

Figure 2: Architecture of 3 layers CNN Model



C. Transfer Learning using the ResNet-50 Model

There are a few variants of the original ResNet architecture that differ by the number of layers used. A deeper network with more layers will achieve a better accuracy score at the expense of more computation time required (as a deeper network will have more Floating-point operations – FLOP) [6].

ResNet-50 is selected as it achieves a good balance between the computation required and the accuracy score achieved. ResNet-50 can achieve a significantly better accuracy score compared to the next shallower model ResNet-34 while requiring about the same amount of computation [6]. On the other end, although the next deeper ResNet-100 can achieve a better accuracy score, it requires almost double computations compared to ResNet-50 [6].

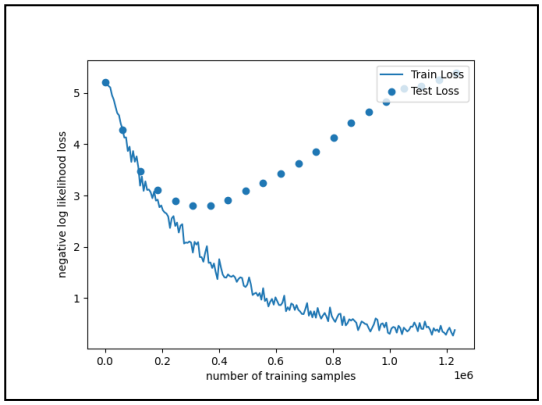
To save the training time, instead of training the ResNet50 from scratch to classify the plant species based on their leaves, a pre-trained ResNet-50 model was used. All the parameters of the pre-trained ResNet-50 model except for the last fully connected layer were frozen during the fine-tuning process. The last fully connected layer is replaced with a new fully connected layer connecting the 2048 output nodes from the previous layer to the 185 nodes for the final classification. Log SoftMax function is applied to enable ranking of the top 5 predictions to obtain the top 5 accuracy scores.

IV. Experiments and Results

A. With The Baseline Model (2CNN)

The baseline model was trained on three different subsets of images. The first subset involved the leave images taken from the field after the segmentation, as shown in Table 1 the top 5 accuracy score achieved after 100 epochs of training is only 68.33% (Table 1). This is probably due to the small number of the training samples being used. As the model tried to fit the training data, the variance (difference between the training loss and testing loss) started to become wider after 300k training samples (see Figure 2) and the test losses increased, indicating that overfitting had occurred.

Figure 3: Training loss and Test loss for the first run



Hence, the second subset used the original leave images taken from the lab condition (without segmentation) to increase the number of training samples to about five times more than the previous run. As a result, the top 5 accuracy improved to 97.75% and the model was able to reach the accuracy desired after 60 epochs of training (Table 1).

However, noticed that the lab images contain the color palette sidebar (see Figure 3), and we would like the model to be able to recognize field images, the baseline model was retrained using the third subset of images which includes all the original leave images from the lab and the field. The top 5 accuracy score drops to 91.25% (Table 1), potentially due to the mixing of the lab images with the field images. A closer look at the variance shows that the training loss and testing loss start to get wider after 50 epochs (figure 4), indicating that overfitting might have occurred.

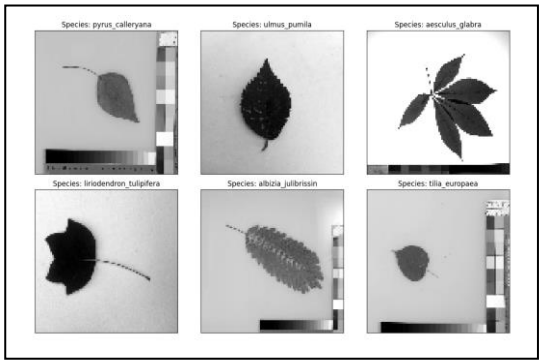
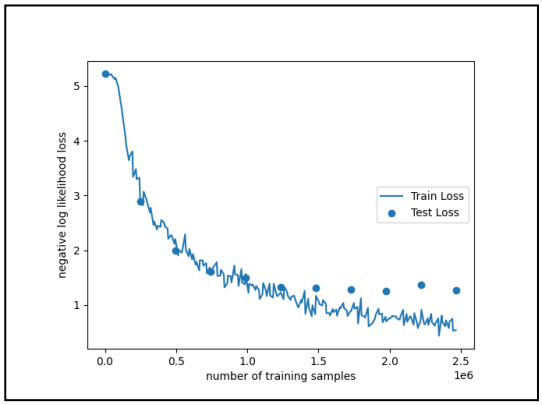


Figure 4: Example Images used for base run. Only images at the top middle and bottom left are from the field, the rest are from the lab

Figure 5: Training loss and Test loss for the third run



B. With The Baseline Model (2CNN) – Variation attempt

Attempts were made to improve the results while maintaining the baseline model and the training dataset used in the third run. Two main approaches have been tried which are increasing the size of the input images and using the original depth (3 RGB channels) of the image.

For the first approach, the input size of the image had been increased from 44x44 during the previous run to 224x224 (the same input size used by the ResNet50). As shown in Figure 5, this results in the overfitting occurring much earlier compared to the third run after only 10 epochs of training. This leads to a drop in the top 5 accuracy score of -1.89% to 89.39% (Table 1)

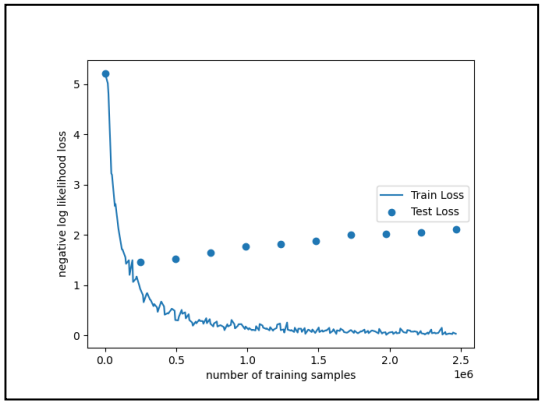
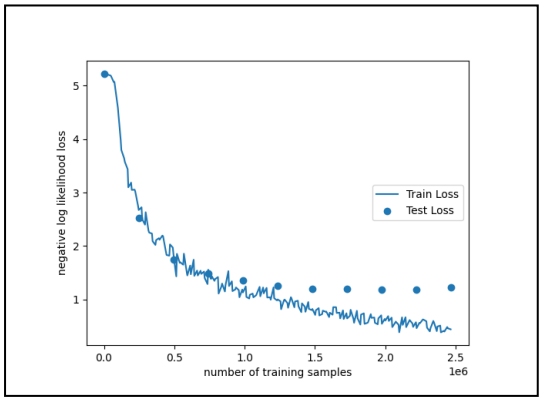


Figure 6: Losses for baseline model with large input size

The second approach tries to pass the original image RGB values of 3 channels to the baseline model for training and testing instead of converting it to grayscale in the previous run. Compared to the third run, the top five accuracy scores improved by 1.61% to 92.86%. This is likely to be the maximum that can be achieved by this approach given that the variance starts to become wider after 50 epochs (Figure 6).

Figure 7: Losses for baseline model with RGB channels

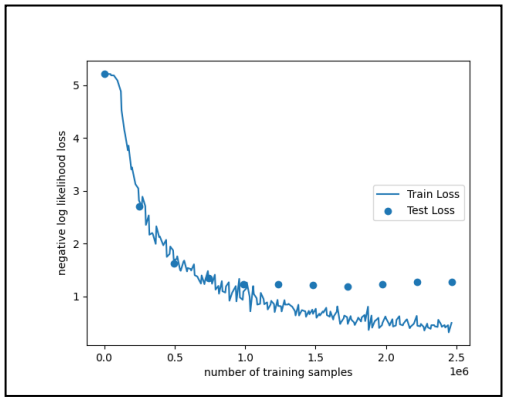


C. With The Deeper Model (3CNN)

This experiment simply uses the deeper model (3CNN) described before and runs against the full data set. All images are converted to the grayscale and resized to 92 x 92 before input into the model.

Compared to the baseline model result in the third run, the top 5 accuracy scores for this network architecture improved by 1.23% to 92.48%. Note that the input image is still converted to grayscale. Judging from the training losses and test losses curve (Figure 8) this is likely to be the maximum achievable as well as the variance starts to get wider after 50 epochs of training.

Figure 8: Losses for the model with 3 CNN stacks



D. With The ResNet-50 model

Two experiments are being conducted. The first experiment tries to fine-tune the model using the full dataset mixing the original images from the lab and field. The images were fed into the model in the original RGB channel. Figure 9 shows some of the example images used.

The model achieves top 5 accuracy scores of 98.1% and reaches the desired accuracy level within 50 epochs of training. The top 1 accuracy score of 87.64% is also much better compared to the previous model of simpler architecture (Table 1).

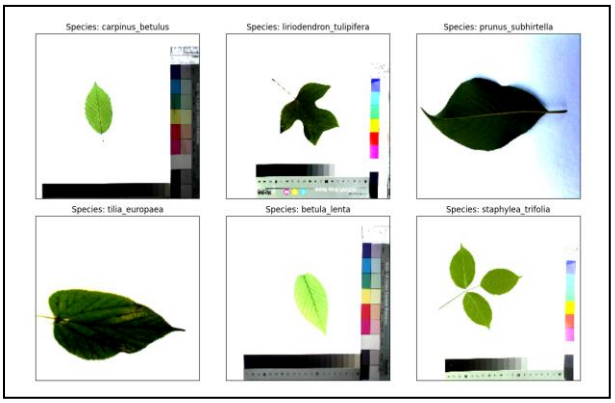
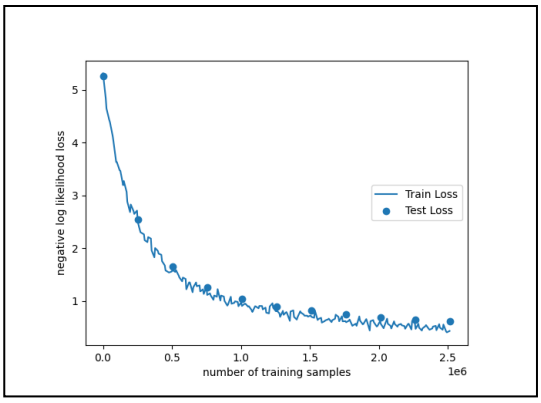


Figure 9: Examples of images used for Resnet training. Top right and bottom left images are from the field, while the rest are from lab.

The second experiment tries to incorporate the classification step to determine if the image contains a leaf of a particular species before proceeding. This is conducted by adding another class consisting of all images without leaves. As shown in Table 1, the top five accuracy score achieved is 98.32% with the desired accuracy target achieved within 50 epochs of training. The training losses and validation losses curve is shown in Figure 10. As both losses still decrease near the end of training and the variance doesn't get wider, it is believed that better results can be achieved with a greater number of epochs for training.

Figure 10: Losses for ResNet-50 model



V. Discussion and Summary

A. Summary of the Results

Table 1 summarizes the test results for all the models trained in the previous sections. As mentioned before, a pre-trained model of ResNet-50 achieved the best top 1 and top 5 accuracy scores compared to the rest and is deemed good enough for our application. Compared with the original recognition system built by the Leafsnap team, this model achieved a better accuracy score and offers the advantage of combining multiple steps.

A program was built to utilize the trained model for the classification of the new images. Figure 11 shows the classification results for some of the test samples. The computation time taken to classify 6174 samples using CUDA-enabled GPU is 167 seconds, which is an average of about 0.027 seconds per image. However, when trying to classify a smaller number of samples (of 439 images), the

average computation time drops to 0.0094 seconds, indicating that 0.027 seconds per image is likely the upper limit in practical use cases.

TABLE I. COMPARISON OF TRAINING RESULTS FOR DIFFERENT MODELS & DATASETS

Model	Dataset	Results		
		Top1 [a]	*Top5* [b]	*No epoch*[c]
2CNN	Segmented Field[d]	37.82%	68.33%	N/A
2CNN	Original Lab[e]	77.62%	97.75%	60
2CNN	Original Lab + Field[f]	68.53%	91.25%	N/A
2CNN (Large)[g]	Original Lab + Field	70.10%	89.39%	N/A
2CNN (RGB)[h]	Original Lab + Field	70.96%	92.86%	N/A
3CNN[i]	Original Lab + Field	71.91%	92.48%	N/A
ResNet-50	Original Lab + Field	87.64%	98.10%	50
ResNet-50	Original Lab + Field + Non-Leaf [j]	88.29%	98.32%	50

[a.] For target within top1 of predicted output.

[b.] For target within top5 of predicted output.

[c.] Number of epochs to reach target top5 score of > 96.8%

[d.] Leaves from the field after the segmentation step

[e.] Leaves from the lab in the original RGB channel

[f.] Leaves from the lab and field in the original RGB channel

[g.] Base model with larger input size (of 224 x 224) to compare with ResNet-50

[h.] Base model with RGB input channel to compared with ResNet-50

[i.] The model with 3 convolution stacks

[j.] Non-leaf consists of random images

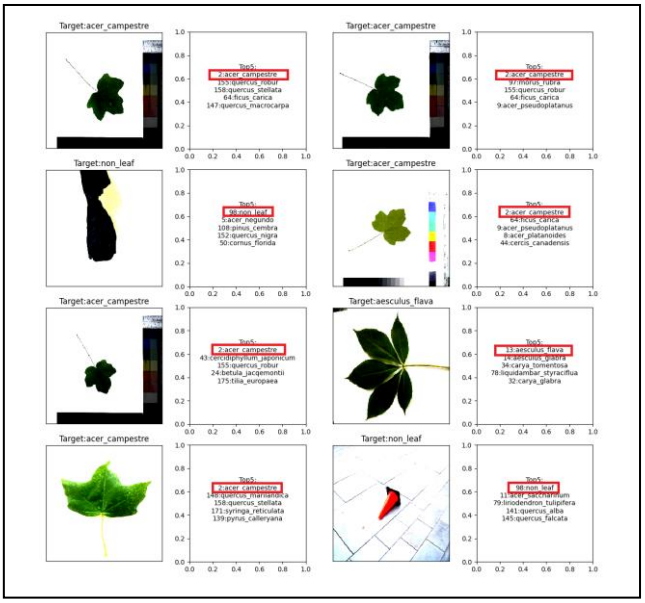


Figure 11: Example classification results using ResNet-50 including non-leaf images.

B. Conclusion and Next Steps

Based on the results, it can be concluded that the Convolution Neural Network based on the ResNet-50 model is more accurate as it was able to achieve a top5 accuracy score of 98.10% (1.3% better than the Leafsnap system) with the potential to obtain better score with more training epochs. The model also has the potential to be faster in the time required for prediction, with the current computation a fraction (< 0.01) of the computation time required for the Leafsnap system. The computation time is also expected to remain stable when the number of training samples increases, whereas, for the Leafsnap system, the computation time will scale up with increasing sample sizes.

Besides, it is also simpler to build and train as it combines the four-step process of the Leafsnap system into one, letting the model learn how to classify leaf / non-leaf and perform segmentation between the leaf and background. This helps get rid of the search for an accurate classification model and proper segmentation algorithm, and the associated errors associated with them.

The next step will be trying to put the system into production to predict plant species based on real-life images. This involves building a mobile application allowing the user to capture the leaf images on the field and send them to the backend server for classification. The backend server will host the trained machine learning model, predict the potential matches based on the query image, and return the top predictions to the mobile application. The mobile application can then display the top 5 results with example images for selection by the user.

ACKNOWLEDGMENT

This work was supervised by Professor Bruce Maxwell from Khoury College, Northeastern University, Seattle Campus.

REFERENCES

[1] https://github.com/Greenstand

[2] Kumar, N. et al. (2012). Leafsnap: A Computer Vision System for Automatic Plant Species Identification. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds) Computer Vision – ECCV 2012. ECCV 2012. Lecture Notes in Computer Science, vol 7573. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-33709-3_36

[3] Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, Liu T, Wang X, Wang G, Cai J, et al. Recent advances in convolutional neural networks. Pattern Recogn. 2018;77:354–77.

[4] H. Ling and D. W. Jacobs, "Shape Classification Using the Inner-Distance," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 2, pp. 286-299, Feb. 2007, doi: 10.1109/TPAMI.2007.41.

[5] Belhumeur, P.N. et al. (2008). Searching the World's Herbaria: A System for Visual Identification of Plant Species. In: Forsyth, D., Torr, P., Zisserman, A. (eds) Computer Vision – ECCV 2008. ECCV 2008. Lecture Notes in Computer Science, vol 5305. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-88693-8_9

[6] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[7] Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition.