# Object identification and classification using Neural Network

**Deven Chhetri** DCHHETRI@UNCC.EDU

**Naman Verma** NVERMA4@UNCC.EDU

**Hemanth Sai Thota** HTHOTA@UNCC.EDU

UNC Charlotte, Charlotte NC

## Abstract

The goal of the project is to design and implement a machine learning algorithm that does some part of human visual system. We implement two neural networks that takes an image as an input. The first network segments the image using conventional encoders and decoders to identify objects present in the image, which is then passed to the second network which is a pre-trained convolutional neural network that provides a summary of the objects, class label and the number of objects, as output.

## 1. Introduction

The importance of image segmentation and object recognition is an important and standard challenge in machine learning and computer vision, thereby being an active area of research for several decades. Image segmentation is the process of partitioning a digital image into multiple segments by clustering pixels into salient image regions and Object recognition is a process for recognizing a specific object in a digital or a video. Combining object recognition and low level segmentation has been shown to improve segmentation precision. Nevertheless, the effects of image segmentation as a pre-processing step for object recognition and classification are still unclear.

A motivation for us to pursue this work is the vast arena of opportunities that is using this technology and research. For example, the concept of Amazon Go shopping relies heavily on computer vision and advanced machine learning techniques, primarily object recognition and image segmentation among others. Another example in which image segmentation and object recognition is prominent is its use and research in the field of self-driving cars.

Our starting point is to build a network, namely convolutional neural network train it with an appropriate dataset so that it classifies the images/objects passed to this network into correct labels or at least the label which closely resembles the object//objects in the image. Then we build another network, using auto-encoder, decoder and softmax function, which segments the image into various objects which is passed to the trained convolutional network that classifies the objects into respective labels. The details are discussed in more detail in methodology section.

Basics:

`Convolutional Neural Network:` Convolution neural networks (CNNs or ConvNets) are essential tools for deep learning, and are especially suited for image recognition. A CNN architecture is formed by a stack of distinct layers that transform the input volume into an output volume through a differentiable function. The layers of the convolutional neural network are:

- Convolutional layer: The convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume.

- Pooling layer: Another important concept of CNNs is pooling, which is a form of non-linear down-sampling. There are several non-linear functions to implement pooling among which max pooling is the most common. It partitions the input image into a set of non-overlapping rectangles and, for each such sub-region, outputs the maximum.

- ReLU layer: ReLU is the abbreviation of Rectified Linear Units. This is a layer of neurons that applies the non-saturating activation function . It increases the nonlinear properties of the decision function and of the overall network without affecting the receptive fields of the convolution layer.

- Fully connected layer: Finally, after several convolutional and max pooling layers, the high-level reasoning in the neural network is done via fully connected layers. Neurons in a fully connected layer have full connections to all activations in the previous layer.

`Auto Encoders:`An autoencoder is an artificial neural network used for unsupervised learning. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction, and reconstruction its own inputs. An autoencoder always consists of two parts, the encoder and the decoder.

## 2. Methodology

Here we explain the design and implementation of the networks and the datasets, and the approach that we followed to stick with our final configurations, in detail.

### 2.1. Convolutional Neural Network and its Dataset section:

The dataset that we used for this is CIFAR-100 dataset, which is found at:
https://www.cs.toronto.edu/ kriz/cifar.html
The dataset consists of 60,000 images in binary format (although there were images in .m, or matlab format, we used images in binary format) which is divided into training data: 50,000 images, and testing data: 10,000 images. Also, the the images are classified in two ways:
first is that the images are classified into 100 class labels which is defined in the dataset as fine labels and the second method of classification is into 20 class labels which is defined in the dataset as coarse labels. We can think of the two types of labels in this way: coarse labels is the superclass and fine labels is the subclass, though the classification can be done entirely by using just coarse labels or fine labels.

So, to serve the purpose of both type of labels we build two convolutional networks, one for coarse labels and the other for fine labels, which are trained independently. The convolutional neural network that we implement classifies the objects that is segmented by the other network, which we discuss later.
We model convolutional neural network using Matlab Deep Learning Library with in-built layers that it provides with much ease. This makes our task relatively simple to implement and to experiment with the layers.
For the fine label classification , we proceed with the below configuration after several iterations and experimentation:
Convolution Layer with ReLu activation
Convolution Layer with ReLu activation
Max Pooling Layer
Convolution Layer with ReLu activation
Convolution Layer with ReLu activation
Max Pooling Layer
Convolution Layer with ReLu activation
Max Pooling Layer
Fully Connected Layer
Softmax Layer.

For the coarse label classification, we choose below configuration:
Convolution Layer with ReLu activation
Convolution Layer with ReLu activation
Max Pooling Layer
Convolution Layer with ReLu activation
Max Pooling Layer
Fully Connected Layer
Softmax Layer.

We have chosen the above architectures after umpteen number of experiments. Since convolution layer deals with extraction of features, we can define the filter size and the number of filters for this layer, but it is not that we can include as many number of filters in the convolution layer or define as many convolution layer as might like, as more number of layers or more number of filters lead to huge computation costs and also overfitting of data. The max pool layer has an advantage in reducing the size of the feature maps but it loses some less important feature maintaining the local invariants, so we have maintained an adequate number of max-pool layers as can be seen above. We include more details regarding the model accuracy values for a few architecture that

we tried in the experiment section. The ReLu layer is just an activation function that we use along with the convolution layer.
We have done Data Augmentation and Normalisation of the data to avoid overfitting for this network.

### 2.2. Neural Network for Image segmentation and its Dataset:

The dataset for this network is BSDS500 which is available at:
https://www2.eecs.berkeley.edu/Research/Projects/CS/vi
The dataset consists of 500 images - 200 images for training, 100 images for validation and 200 images for testing.
For this purpose, we build an auto-encoder, which first encodes a given image into features and then decodes it to reconstruct back the image. Here, we collect the features from the encoder, and match it with the corresponding image pixel's labels using soft max layer. These features can be thought of as pixel's descriptor which is used later to classify the segmented objects. and Then, we train this soft max network to learn the labels of the pixels using given features and we build a deep network by combining the encoder and soft max net, and train it on images using pixel labellings from segmentation.
We build a network object for each image as we train the model. So, during testing we pass the image to all of the networks and check the one which gives the highest number of labels (more segmentation) and then we use connected component technique to score each locally available objects in the image.
Based on these scores, we only chose scores representing region of pixels over a certain threshold value and pass it to the trained convolution neural networks, which we have described above in the Convolutional Neural Network and its Dataset section to get the labels for each region. Based on the confidence measure with which the network classifies the region, we accept or reject those labels for the regions.

## 3. Experiment

The following figure(Figure 1) shows the accuracy for three of the convolutional network architectures that we tried. It clearly explain the first few layers as the layers of the network with its corresponding parameters. The options state the epoch for that particular experiment, and the initial learning rate is the rate at which the optimising function alters the weights.

```
1)convolution2dLayer([5 5],20,'Stride',1,'Padding',2);
            reluLayer();
            maxPooling2dLayer(2,'Stride',2);
            convolution2dLayer([5 5],20,'Stride',1,'Padding',2);
            reluLayer();
            maxPooling2dLayer(2,'Stride',2);
            fullyConnectedLayer(200);
            reluLayer();
            fullyConnectedLayer(100);
options = trainingOptions('MaxEpochs', 50,'InitialLearnRate', 0.01);
Accuracy = 15

2)same convnet as above
options = trainingOptions('MaxEpochs', 50,'InitialLearnRate', 0.001);
Accuracy = 75

3)imageInputLayer([32 32 3]);
            convolution2dLayer([3 3],32,'Stride',1,'Padding',2);
            reluLayer();
            maxPooling2dLayer(2,'Stride',2);
            convolution2dLayer([3 3],64,'Stride',1,'Padding',2);
            reluLayer();
            maxPooling2dLayer(2,'Stride',2);
            convolution2dLayer([3 3],128,'Stride',1,'Padding',2);
            reluLayer();
            maxPooling2dLayer(2,'Stride',2);
            fullyConnectedLayer(100);
options = trainingOptions('MaxEpochs', 40, 'InitialLearnRate', 0.001);
Accuracy = 90

4)Chosen CNN model
Training Accuracy = 87          Testing Accuracy=32
```

*Figure 1.* Classification Accuracies for three CNN architectures

## 4. Results

The Convolution neural network works perfectly fine, although with less accuracy of 32 percent on testing data for fine labels and 45 percent for coarse labels.
The Network for image segmentation works well but we are not able to gauge the accuracy of the model as we are still not completely correct with the way we are passing the segmented features to the convolutional neural network. The images that have features which have more of global presence works well with coarse labels and those with finer details work good with fine labels. The below(Figure 2) is the segmented image for horse from bsds dataset.

The following figure shows the fine label classification, coarse label classification and the number of objects detected in the given image (Figure 3)

*Figure 2.* Segmented images of horse

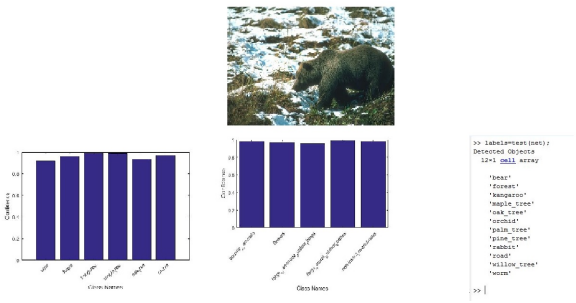


*Figure 3.* Object predictions for bear image

## 5. Discussion and Future Work

There are so many edges from which we can continue working on this project. Firstly, we can experiment more with parameters and hyper-parameters for building convolutional neural network and auto-encoders. We can fine tune the output from Image segmentation neural network to be able to properly feed it into Convolution neural network. We could also model the convolution neural networks for fine labels and coarse labels in such a a way that a single model could handle both of these classification.

In image segmentation section, here we are building a network for each image. In future, we want to improve on this model and just build a single network which could handle the combined networks for all these networks for the images. Also, the images having more global features and more finer features are classified in dissimilar manner, like the ones with more global features are classified well with coarse labels and the ones with finer details are classified better with fine labels. So, we want to improve on this by building a single model which does both of this tasks effectively. Lastly, we can explore various deep learning libraries that are available for this type of work, though we tried our hands on Tensor Flow in this short time, but could not achieve the desired results with it.

## 6. References

1.Jonathan Long, Evan Shelhamer, Trevor Darrell: Fully Convolutional Network for Semantic Segmentation, cs.berkeley.edu

2.Review on Image Segmentation, Clustering and Boundary Encoding, International Journal of Innovative Research in Science, Engineering and Technology (An ISO 297: 2007 Certified Organization), Vol. 2, Issue 11, November 2013

3.Girshick, R., Donahue, J., Darrell, T., Malik, J: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)

4.Shawn McCann, Jim Reesman: Object Detection using Convolutional Neural Networks

5.B Hariharan, P Arbel, R Girshick, and J Malik : Simultaneous Detection and Segmentation. In CVPR 2015

6.V Badrinarayan, A Kendall, R Cipolla; "SegNet: A Deep Convolutional Encoder - Decode Architecture for Image Segmentation", arXiv:1511.00561v3, 2016