Birla Institute of Technology & Science, Pilani

CS F366 - Laboratory Project

Object Detection using Deep Learning framework

Guided By:

Ms. Somsukla Maiti

Scientist, CSIR-CEERI

Submitted By:

M Sharat Chandra - 2014A7TS0108P

Harshvardhan Maheshwari - 2014B3A70375P

April 27, 2018

Table of Contents

1. Introduction

Human beings are able to detect visually distinctive, so called salient, scene regions effortlessly and rapidly (i.e., preattentive stage). These filtered regions are then perceived and processed in finer details for the extraction of richer high-level information (i.e., attentive stage). This capability has long been studied by cognitive scientists and has recently attracted a lot of interest in the computer vision community mainly because it helps find the objects or regions that efficiently represent a scene and thus harness complex vision problems such as scene understanding.

Saliency measures the low-level stimuli to human vision, and serves as an alternative to semantic image understanding. Some topics that are closely or remotely related to visual saliency include: salient object detection, fixation prediction, object importance, memorability, scene clutter, video interestingness, surprise, image quality assessment, scene typicality, aesthetic and attributes. Given space limitations, this paper cannot fully explore all the aforementioned research directions. Instead, we only focus on salient object detection, a research area that has been greatly developed in the past twenty years in particular since 2007.

In general, it is agreed that for good saliency detection a model should meet at least the following three criteria: 1) good detection: the probability of missing real salient regions and falsely marking the background as a salient region should be low, 2) high resolution: saliency maps should have high or full resolution to accurately locate salient objects and retain original image information, and 3) computational efficiency: as front-ends to other complex processes, these models should detect salient regions quickly.

In this project, our focus is to detect salient objects in the images taken by a UAV/MAV. Fast and accurate vehicle detection in unmanned aerial vehicle (UAV) images remains a challenge, due to its very high spatial resolution and very few annotations. Although numerous vehicle detection methods exist, most of them cannot achieve real-time detection for different scenes. Extracting and identifying vehicles in unmanned aerial vehicle (UAV) images plays an important role for a wide range of applications and is receiving significant attention in recent years. However, it is still a challenging problem due to the high resolution with extremely high level of detail, various shooting platform, limited annotated data, and limited processing time for real-time applications.

2. Background

2.1 Object Detection

“Salient object detection” or “salient object segmentation” is commonly interpreted in computer vision as a process that includes two stages: 1) detecting the most salient object and 2) segmenting the accurate region of that object. Rarely, however, models explicitly distinguish between these two stages (with few exceptions). Following the seminal works by Itti et al. [16] and Liu et al. [17], models adopt the saliency concept to simultaneously perform the two stages together. This is witnessed by the fact that these stages have not been separately evaluated. Further, mostly area-based scores have been employed for model evaluation (e.g., Precision-recall). The first stage does not necessarily need to be limited to only one object. The majority of existing models, however, attempt to segment the most salient object, although their prediction maps can be used to find several objects in the scene. The second stage falls in the realm of classic segmentation problems in computer vision but with the difference that here accuracy is only determined by the most salient object.

Salient object detection models usually aim to detect only the most salient objects in a scene and segment the whole extent of those objects. Fixation prediction models, on the other hand, typically try to predict where humans look, i.e., a small set of fixation points. Since the two types of methods output a single continuous-valued saliency map, where a higher value in this map indicates that the corresponding image pixel is more likely to be attended, they can be used interchangeably.

One of the earliest saliency models, proposed by Itti et al. [16], generated the first wave of interest across multiple disciplines including cognitive psychology, neuroscience, and computer vision. This model is an implementation of earlier general computational frameworks and psychological theories of bottom-up attention based on center-surround mechanisms (e.g., Feature Integration Theory by Treisman and Gelade, Guided Search Model by Wolfe et al., and the Computational Attention Architecture by Koch and Ullman). In [16], Itti et al. show some examples where their model is able to detect spatial discontinuities in scenes. Subsequent behavioral and computational investigations used fixations as a means to verify the saliency hypothesis and to compare models.

2.2 Deep Learning

Deep learning is being used in the research community and industry to help solve many big data problems such as computer vision, speech recognition, and natural language processing. Using advanced neural networks algorithms, Big Data, and the computational power of the GPU, machines are now able to learn it at a speed, accuracy, and scale that are driving true artificial intelligence [8]. Neural Networks are a beautiful biologically-inspired programming paradigm which enables a computer to learn from observational data. Deep Learning is a powerful set of techniques for learning in neural networks. [7]

The Deep Learning model is a feed-forward neural network. Each of the sparse, high-dimensional categorical features are first converted into a low-dimensional and dense real-valued vector, often referred to as an embedding vector. These low-dimensional dense embedding vectors are concatenated with the continuous features, and then fed into the hidden layers of a neural network in the forward pass. The embedding values are initialized randomly, and are trained along with all other model parameters to minimize the training loss.

Feedforward neural networks are called networks because they are typically represented by composing together many different functions. The model is associated with a directed acyclic graph describing how the functions are composed together. [1]

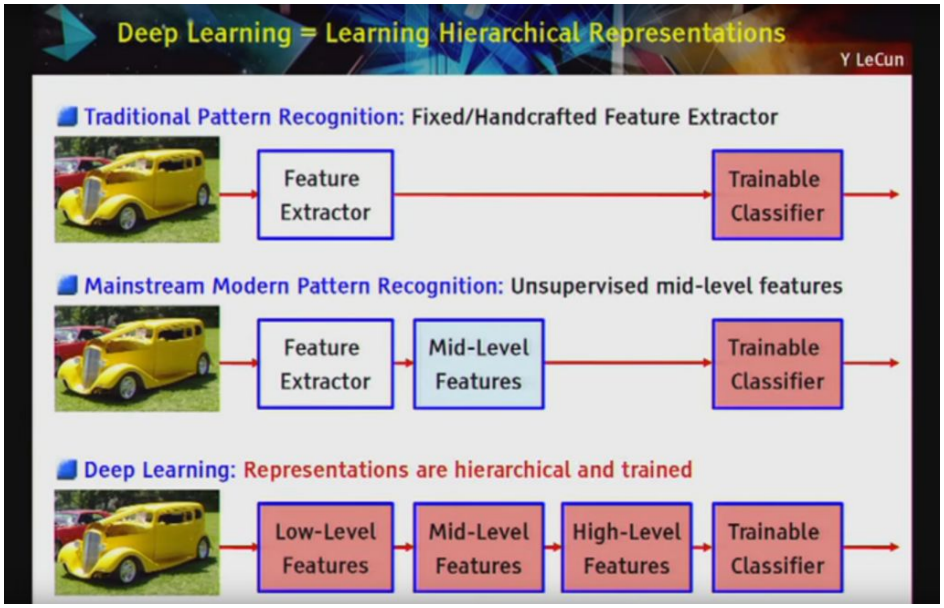

Figure : Understanding Deep Learning Approach

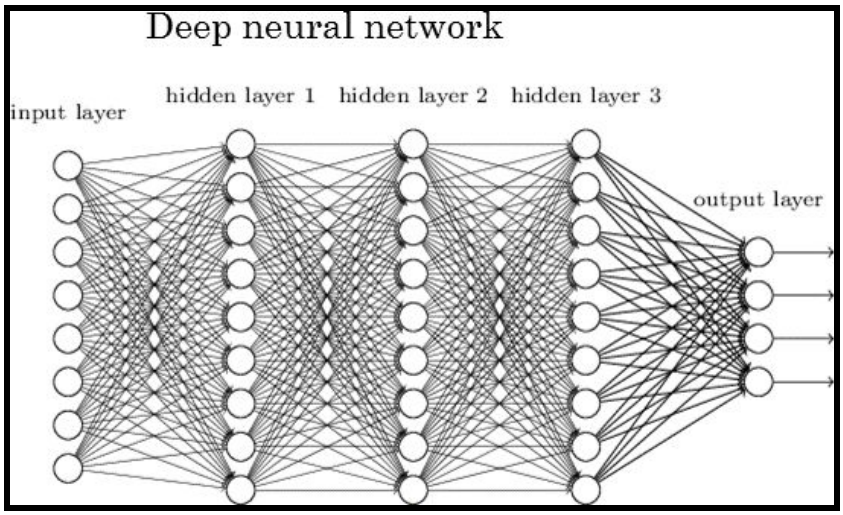

Figure : Conceptual idea about Deep Neural Network [7]

2.2.4 Convolutional Neural Networks (CNN)

2.2.4.1 Overview

Convolutional neural networks or CNNs, are a specialized kind of neural network for processing data that has a known, grid-like topology[3]. CNNs are quite similar to normal Neural Networks. They are made up of neurons having learnable weights and biases. However, CNN architecture makes the explicit assumption that the inputs are images. This enables encoding of certain special properties into the architecture. These, in turn, make the forward function more efficient to implement.

Thus, these networks use a special architecture which is particularly well-adapted to classify images. Using this architecture makes convolutional networks fast to train. This, in turns, helps us train deep, many-layer networks, which are very good at classifying images. Today, deep convolutional networks or some close variant are used in most neural networks for image recognition.[7]

Small (often minimal) receptive fields of convolutional winner-take-all neurons yield large network depth, resulting in roughly as many sparsely connected neural layers as found in mammals between retina and visual cortex. Only winner neurons are trained. Several deep neural columns become experts on inputs preprocessed in different ways; their predictions are averaged. Graphics cards allow for fast training. [4]

2.2.4.2 Architecture

The architecture of a typical CNN is composed of multiple convolutional and subsampling layers. Each layer perform a specific function to transform its input into more useful representation.

Input to Convolutional Layer A ***m x m x r image*** where ***m*** is the height and width of the image and ***r*** is the number of channels.

Within the Layers A number of filters (or Kernels) of size ***n x n x q*** where ***n*** is smaller than the dimension of the image. The size of the filters gives rise to the locally connected structure which are each convolved with the image to produce ***k*** feature maps of size ***m-n+1***.

Each map is subsampled typically with **mean** or **max pooling** over ***p x p*** contiguous regions, where ***p*** ranges between 2 (for small images) to 5 (for larger inputs). An additive bias and sigmoidal nonlinearity is applied to each feature map. The figure below describes a 2D CNN with Convolutional and Subsampling Layers. The units with same color have their weights tied. After convolutional layers, there are many fully connected layers, which appear identical to standard multilayer neural network.[11]

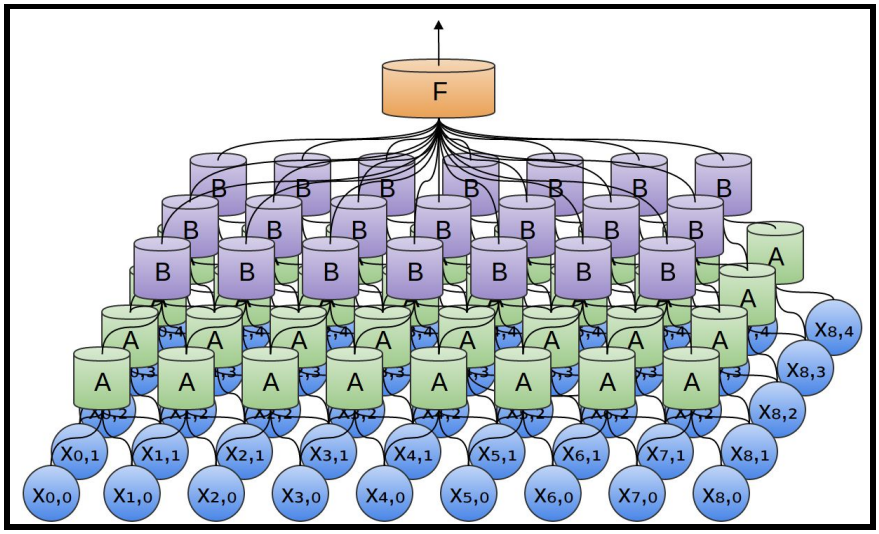

Figure:A 2D Convolutional Neural Network with Convolutional & Subsampling Layers [11]

2.2.5 Merits and Demerits of Neural Networks

Advantages:

- Neural Networks have high tolerance for noisy data.
- They are also inherently parallel, which makes them well-suited for continuous valued inputs-outputs.

- This technique can be used even when we have little knowledge about relationship between attributes and classes

Disadvantages:

- Long training times which can be overcome by techniques like convolutions, pooling, the use of GPUs to perform parallel computations, the algorithmic expansion of our training data (to reduce overfitting), the use of the dropout technique (also to reduce overfitting), the use of ensembles of networks, and others.
- Poor interpretability of the functions performed in the hidden layers.

3. Approaches to object detection

3.1 Traditional approach

A multi-stage salient detection system is proposed which combines low-level contrast features, mean shift segmentation with additional histogram information and multichannel edge features gathered over several feature maps. Figure below presents a block diagram of proposed saliency detection architecture in which we see the various steps of feature detection and amalgamation.

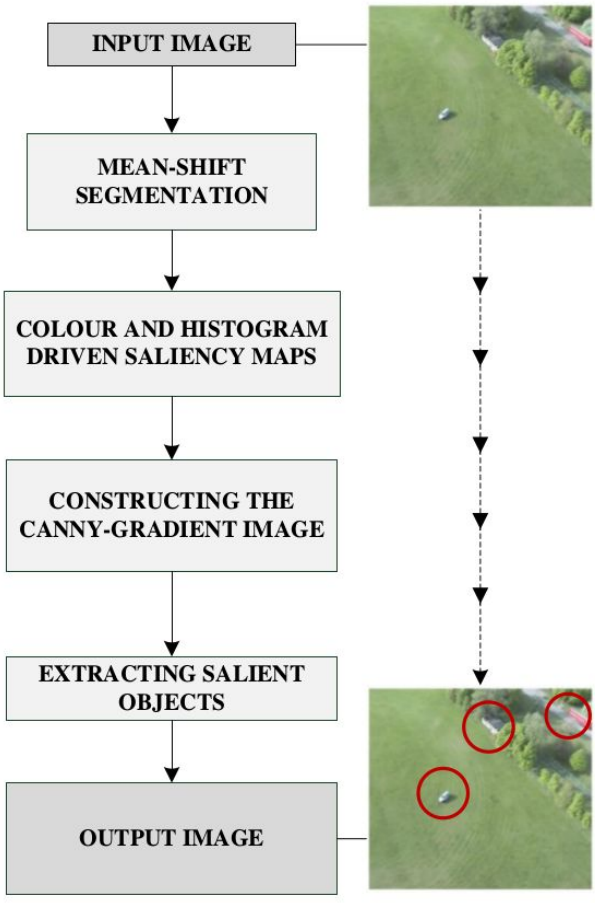

Figure: Block diagram of traditional approach

3.1.1 Mean-shift Segmentation

Using mean-shift segmentation with low-level contrast features has proven to add significant information about the saliency of image regions. Here we follow this approach as using mean-shift segmentation with appropriate radii in both the spatial and colour domains has the ability to preserve small entities in image whilst at the same time merging larger, uniform image regions.

3.1.2 Colour and Histogram driven Saliency Maps

The mean-shift segmented image (Section 4.1) is first transformed into the LUV colour space (i.e. L* u* v* (6)) and a two level Gaussian image pyramid (6) is created out of which the final contrast map is derived. We alternatively calculate contrast as a Euclidean distance which is a characteristic feature of the LUV colour space, as shown in the below equation:

$$C_{i,j,l} = \sum_{q \in \Theta} d(p_{i,j,l}, p_q)$$

Where in the above equation, $C_{i,j,l}$ is accumulated contrast value for every pixel, p_q, in theta-sized neighborhood of pixel (i,j), $p_{(i,j,l)}$, at level l within the pyramid. The contrast maps from each level within the pyramid are then rescaled to the original image size and additively combined into one image. At this point of recombination we also add additional information about colour distribution from the HSV (Hue, Saturation and Variance colour space) histogram of the mean-shift image. Each contrast pixel value is effectively multiplied by an inversely weighted probability of its occurrence in a normalized two- dimensional histogram of hue and saturation (H and S channel histogram of HSV image). Using this information each contrast pixel value in the final resized contrast map is suppressed if its frequency is high within the histogram distribution whilst those which appear with limited frequency are enhanced. Thus contrast pixel values with an associated high probability of hue and saturation occurrence within the image are effectively suppressed whilst the less probable contrast pixel values within the image are enhanced. In this way we combine both of the traditional contrast map based approaches for saliency detection (13) with the additional enhancement of likelihood of occurrence probability of a given colour within the image.

3.1.3 Constructing the Canny-Gradient Image

In this next stage we gather multiple sources of edge information into two separate images, one for Canny edges and one for gradient edges, where the latter are defined with the use of morphological operation of erosion and dilation as follows:

gradient(i) = dilate(i) – erode(i)

Where in the above equation, i is the input image over a set of nine feature images. Overall edge information is gathered on a set of nine feature images as follows: the mean-shift image, the contrast map with added histogram information and seven feature images obtained following Itti and Koch's original seminal method. These seven feature images are three original color channels (R,G,B) normalized by intensity and four broadly tuned color channels.

$$intensity = \frac{(r + g + b)}{3}$$

In addition the broadly tuned channels, as defined by Itti and Koch [19] can be expressed as follows:

$$R = r - \frac{(g + b)}{2}$$

$$G = g - \frac{(r+b)}{2}$$

$$B = b - \frac{(g + r)}{2}$$

$$Y = \frac{(r + g) - |r - g|}{2}$$

Once the set of nine feature images is composed in this manner we additively gather edge information over the entire set into two resulting images, by using both the Canny edge detection operator and gradient edge information. Each of the resulting images is obtained as the sum of responses over each of these operators (Canny and gradient) respectively. In the final step these two resulting images are multiplicatively combined leaving only the most conspicuous edges.

3.1.4 Extracting Salient Objects

The constructed Canny-gradient image is now processed using dilation and connected components to extract coherent salient objects within the scene. Using prior morphological dilation will merge separated objects within the Canny-gradient image prior to further connectivity processing. Currently, the only drawback is occasional false positive detection due to the noise in the environment.

3.1.5 Results

3.2 Deep Learning (YOLO) approach

3.2.1 Network Architecture

The CNN algorithm presented here is based on an open-source object detection and classification platform compiled under the “YOLO” project, which stands for “You Only Look Once” [20]. The “YOLO” has many advantages over other traditionally employed convolutional neural network software. For example, many CNNs use regional proposal methods to suggest potential bounding boxes in images. This is followed by bounding box classification and refinement and the elimination of duplicates. Finally, all bounding boxes are re-scored based on other objects found in the scene. The issue with these methods is that they are applied at multiple locations and scales. High scoring regions of an image are considered to be detections. This procedure is repeated until a certain detection threshold is met. While these algorithms are precise and are currently employed in many applications, they are also computationally expensive and almost impossible to optimize or parallelize. This makes them unsuitable for autonomous UAV applications. On the other hand, “YOLO” uses a single neural network to divide an image into regions, while predicting bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. The main advantage of this approach is that the whole image is evaluated by the neural network,and predictions are made based on the concept of the image, not the proposed regions.

The “YOLO” approaches object-detection as a tensor-regression problem. The procedure starts by inputting an image into the network. The size of the image entering the network needs to be in fixed format (n × m × 3, where the number 3 denotes 3 color channels). Our preliminary results show that the best-performing image size is 448 × 448; therefore, we used a 448 × 448 × 3 format in all tests. Following image formatting, an equally sized grid (S × S) is superimposed over the image, effectively dividing it into N number of cells. Each grid cell predicts the number of bounding boxes (B) and confidence scores for those boxes.

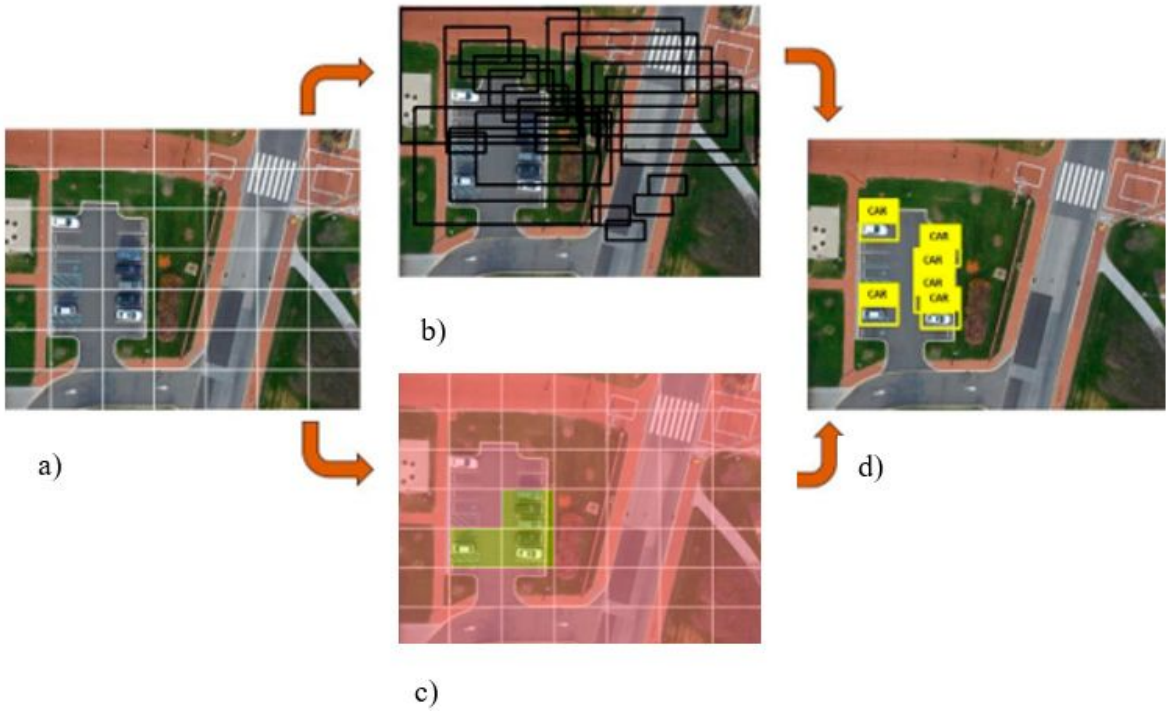

Figure: Images captured by UAVs (a) divided into cells using an equally sized grid (b,c) to uncover key features in the underlying landscape (d). The eg. shows the network designed with grid size S=7 and no. of cells N=49

At this point, each bounding box contains the following information: x and y coordinates of the bounding box, width (w), height (h), and the probability that the bounding box contains the object of bounding box, width (w), height (h), and the probability that the bounding box contains the object interest (Pr (Object)). The (x, y) coordinates are calculated to be at the center of the bounding box but of interest (Pr (Object)). The (x, y) coordinates are calculated to be at the center of the bounding box relative to the bounds of the grid cell (Figure 1c). The width and height of the bounding box are but relative to the bounds of the grid cell (Figure 1c). The width and height of the bounding box are predicted relative to the whole image. The final output of the process is S × S × (B × 5 + C) tensor, predicted relative to the whole image. The final output of the process is S × S × (B × 5 + C) tensor, where C stands for the number of classes the network is classifying and B is a number of hypothetical where C stands for the number of classes the network is classifying and B is a number of hypothetical object bounding boxes. Non-maximal suppression is used to remove duplicate detections. During the object bounding boxes. Non-maximal suppression is used to remove duplicate detections. During the network training phase, the following loss function was implemented: network training phase, the following loss function was implemented:

$$\lambda_{\text{coor}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2$$

where w_i is if the of the bounding box, in while h_i responsible is the height the bounding 1_{ij}^{obj} is the function that counts if the jth bounding box predictor in cell i is responsible for the prediction of the object.

The proposed detection network has only 24 convolutional layers followed by two fully-connected layers. This condensed architecture significantly decreases the time for the object detection, while marginally reducing the classification accuracy of detected objects. THe 26-layer configuration shown in figure is preferred for UAV applications due to its high computational speed. According to the “YOLO” authors, alternating 1 x 1 convolutional layers reduces the feature space from that of the preceding layers. The final layer makes object classification supplemented by the probability that the selected object belongs to the class in question and the bounding box coordinates. Both the bounding box height (h) and width (w); and x and y coordinates, are normalized to have values between 0 and 1.

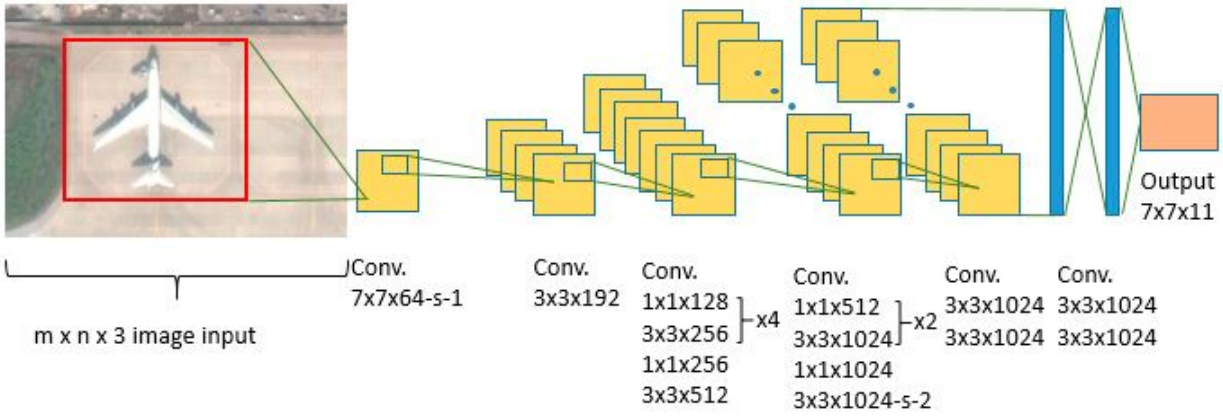

Figure: Graphical representation of multilayered convolutional neural network (CNN) architecture

3.2.2 Dataset

VEDAI [21] is a dataset for Vehicle Detection in Aerial Imagery, provided as a tool to benchmark automatic target recognition algorithms in unconstrained environments. The vehicles contained in the database, in addition of being small, exhibit different variabilities such as multiple orientations, lighting/shadowing changes, specularities or occlusions. Furthermore, each image is available in several spectral bands and resolutions. A precise experimental protocol is also given, ensuring that the experimental results obtained by different people can be properly reproduced and compared. We also give the performance of some baseline algorithms on this dataset, for different settings of these algorithms, to illustrate the difficulties of the task and provide baseline comparisons.

3.2.3 Network Training

While “YOLO” provides a platform for object detection and classification, the CNN still needs to be trained and the correct parameters need to be determined. The batch size, momentum, learning rate, decay, iteration number, and detection thresholds are all task-specific parameters (defined by the user) that need to be inputted into the “YOLO” platform. The number of epochs that our network needed to be trained with was determined empirically. “Epoch” refers to a single presentation of the entire data set to a neural network. For batch training, all of the training samples pass through the learning algorithm simultaneously in one epoch before weights are updated. “Batch size” refers to a number of training examples in one forward/backward pass. “Learning rate” is a constant used to control the rate of learning. “Decay” refers to the ratio between learning rate and epoch, while a momentum is a constant that controls learning rate improvement. Our network was designed to have 7 × 7 grid structure (S = 7), and was tested on only one object class; i.e., “vehicle” (C = 1). This network architecture gives an output tensor with dimensions 7 × 7 × 11.

It is important to note that highly utilized and cited image databases such as PASCAL VOC 2007 and 2012, were not used for the training purposes. Preliminary results showed that images taken by UAVs differed significantly from the images available at the PASCAL VOC databases in terms of the scene composition and angle at which images were taken. For example, many images from the PASCAL VOC database were taken from the frontal view, while the images taken by the UAV consist mostly from the top-down view. Therefore, it was not a coincidence that the networks trained on the PASCAL VOC database images alone when tested on UAV acquired video feeds proved to be very unstable with very low recognition confidence (~20%). However, a recognition confidence of 84% was reached when a database containing satellite and UAV-acquired images was used for the training purposes. The learning rate schedule also depended on the learning data set. While it has been suggested that the learning rate rises slowly for the first epochs, this may not be true for our network training. It is known that starting the learning rate at high levels causes models to become unstable. This project provided a unique opportunity to learn how networks behave when exposed to new data sets. To avoid overfitting, dropout and data augmentation were used during network training. For a dropout layer, a rate of 0.5 was used, while for data augmentation, random scaling and translations of up to 35% of the original image size were implemented. Furthermore, saturation and exposure of the image were randomized by up to a factor of 2.5 in the hue saturation value (HSV).

3.2.4 Results


vehicle


vehicle


vehicle
vehicle

4. Discussion and Conclusion

Object detection has been a relatively difficult task to be performed by a machine in comparison to classification of images. Such a task becomes even more difficult when the images taken are from UAVs thereby making the objects smaller and more difficult to detect. In our project, we attempted to explore different algorithms used for object detection. We particularly looked at the best algorithms in two categories: traditional, i.e. using human-crafted techniques of image processing and contemporary machine learning based algorithms.

In the traditional category, we explored a multi-stage salient detection system which combines low-level contrast features, mean shift segmentation with additional histogram information and multichannel edge features -- all of which are image processing algorithms applied on the image -- to perform object detection. This approach was successfully able to isolate objects in an image that were standing out from their background in the spatial and color domain. However, we observed that if such a separation didn't exist the algorithm was unable to detect the presence of a salient object. Furthermore, as the algorithm just distinguishes these objects from their environment it is not able to classify any of the objects and a separate classifier is needed to further tag the detected objects.

To overcome the above-mentioned problems with the conventional approaches, we explored a state-of-the-art object detection method called YOLO that uses lots of tagged image data to learn about the object and its isolation techniques on its own. This algorithm was known to work well on PASCAL and VOC datasets -- these datasets contain tagged images of everyday scenes. Our objective was to implement and tune this algorithm for the task of object detection in images taken by UAVs. We used VEDAI dataset (Vehicle Detection in Aerial Imagery) to train and validate our model. We observed that this neural network based approach has multiple advantages for object detection over the traditional algorithms. Firstly, it is not only able to detect objects but also classify the object into categories. Secondly, it is able to learn about objects which do not effectively stand out from their surroundings. Thirdly, the speed of detection is much higher compared to any traditional algorithms thus making it feasible to be run on live video streams.

Thus, in conclusion, through these explorations we find that deep learning based approaches, specifically a fast ones like YOLO, can be effectively used to detect objects in the aerial images not only in offline processing but also with a live feed that the UAV drones receive. This fast detection can be leveraged to track vehicles, provide help and support if needed, and also extends possibilities of being used by security organizations.

5. References

1. Jiawei Han and Micheline Kamber ***Data Mining concepts and techniques,*** *Elsiever* Second Edition 2002, pp. 328-340
2. Stuart Russell and Peter Norweig ***Artificial Intelligence A modern Approach,*** Second Edition pp 578-589
3. Ian Goodfellow and Yoshua Bengio and Aaron Courville ***Deep Learning*** Book in preparation for MIT Press, Last accessed online at http://www.deeplearningbook.org, on 14.11.2016.
4. Ciresan, Dan, Ueli Meier, and Jürgen Schmidhuber. ***"Multi-column deep neural networks for image classification."*** *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012.
5. Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton ***ImageNet Classification with Deep Convolutional Neural Networks***, Advances in Neural Information Processing Systems, (NIPS 2012), 2012.
6. Koray Kavukcuoglu, Pierre Sermanet, Y-Lan Boureau, Karol Gregor, Michaël Mathieu and Yann LeCun,__***Learning Convolutional Feature Hierarchies for Visual Recognition***, Advances in Neural Information Processing Systems (NIPS 2010), 23, 2010.
7. Michael Nielsen ***Neural Networks and Deep Learning*** Online Resource on http://neuralnetworksanddeeplearning.com/chap6.html, Jan 2016, Accessed on 14.11.2016.
8. Deep Learning Resources https://developer.nvidia.com/deep-learning
9. Denny Britz ***Understanding Convolutional Neural Networks for NLP***; Image online:http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/
10. Ji Wan, Dayong Wang, Steven Chu-Hong Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, Jintao Li: ***Deep Learning for Content-Based Image Retrieval***: A Comprehensive Study. ACM Multimedia 2014: 157-166
11. Andrew Ng, Jiquan Ngiam, Chuan Yu Foo, Yifan Mai, Caroline Suen, Adam Coates, Andrew Maas, Awni Hannun, Brody Huval, Tao Wang, Sameep Tandon, ***Deep Learning Tutorial***, Online:http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/, Accessed on 14.11.2016.
12. Christopher Olah, ***Understanding Convolutional Neural Network***: Internet Resource http://colah.github.io/posts/2014-07-Conv-Nets-Modular/
13. Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, **CIFAR 10 Dataset.** Url: https://www.cs.toronto.edu/~kriz/cifar.html
14. Google Brain Team, ***TensorFlow***: an Open Source Software Library for Machine Intelligence, Online : https://www.tensorflow.org/versions/r0.11/tutorials/index.html, Accessed on: 25.11.2016
15. Fei Fei Li, Justin Johnson, Andrej Karpathy: Stanford CS class ***CS231n Convolutional Neural Networks for Visual Recognition*** Online:http://cs231n.github.io/optimization-1/ Last Accessed: 25.11.2016
16. L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," IEEE TPAMI, no. 11, pp. 1254–1259, 1998.
17. T. Liu, J. Sun, N. Zheng, X. Tang, and H.-Y. Shum, "Learning to detect a salient object," in CVPR, 2007, pp. 1–8.

18. Itti, L., C. Koch, and E. Niebur. “A Model of
19. Saliency-Based Visual Attention for Rapid Scene Analysis.” IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.20, p.1254-1259, November 1998.
20. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection.
21. Vehicle Detection in Aerial Imagery: A small target detection benchmark., Sébastien Razakarivony and Frédéric Jurie, Journal of Visual Communication and Image Representation, 2015