IE685: MSc-PhD Research Project 1

Topic: Weakly Supervised Deep Detection Network

Rupesh Yadav | 21i190004

Supervisor: Prof. Balamurugan Palaniappan

Industrial Engineering and Operations Research
Indian Institute of Technology, Bombay

Autumn, 2022-2023

Acknowledgement

I would like to thank Prof. P. Balamurugan for his guidance and support in this project. His encouragement had enabled me to solve real-word problem. His advice and guidance have been remarkable in my learning. I would also like to thank him for providing me access to the IEOR GPU SERVER without which it would not have been possible to perform experiments.

I would also like to thank Abdullah Dursun whose pytorch implementation of the original paper I have used as reference for my experiments.
((https://github.com/adursun/wsddn.pytorch.git)

Contents

Chapter 1

Introduction

Object detection is a well-known computer vision problem, where one needs to detect or localize presence of objects by making bounding boxes around all the objects present in the image. In object detection a large amount of research has been done till date and is still very active area of research.

It is widely used in computer vision tasks such as image annotation, vehicle counting, activity recognition, face detection, face recognition. It is also used in tracking objects, for example tracking a ball during a football match, tracking movement of a cricket bat, or tracking a person in a video.

There are different kinds of methods for this task. Some of them are those which uses Deep Learning based techniques. In this report, we will discuss about this category of method only.

Fully supervised object detection methods, which require bounding box annotations in the training images, have become state-of-the-art for object detection. However, due to the inconvenience of gathering a large amount of data with accurate bounding box annotations, weakly supervised object detection (WSOD) is recently seeking a lot of attention as it needs only image level labels (not bounding boxes) in the training images.

In 2016, H Bilen, A Vedaldi [1] tried to address this weakly supervised object detection problem. They exploited the power of deep convolutional neural networks pre-trained on large-scale image-level classification tasks. They build their "Weakly Supervised Deep Detection Network (WSDDN)" as an architectural modification of one such network which achieved wide recognition and state-of-the-art performance at that time.

Chapter 2

Literature Overview

2.1 Neural networks and Deep Learning

Neural networks are inspired by the structure and the functioning of the information processing units, neurons, present in the human brain. It is one of the most beautiful programming paradigms ever invented. In the conventional programming approach we tell the computer to follow a set of pre-defined steps. By contrast, in a neural network we don't tell the computer how to solve our problem. Instead, it learns from observational data, figuring out its own solution to the problem at hand.

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers, in other words a Neural Network with more than one hidden neurons.

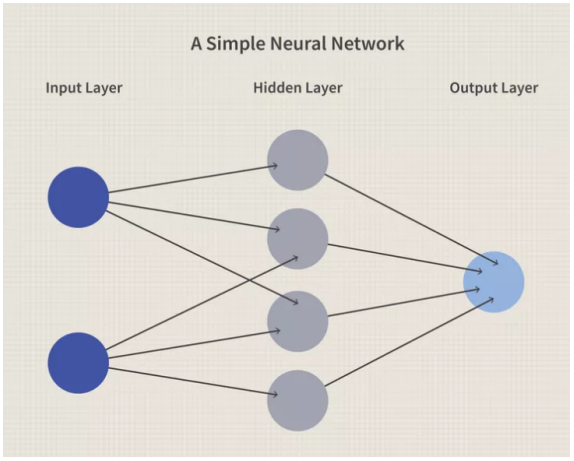

Figure 2.1: A Neural Network [2]

2.2 Feed Forward Neural Network

A feed forward neural network (fig:2.1) was the first and simplest type of artificial neural network in which the information is processed in only one direction, i.e. forward direction - from the input nodes, through the hidden nodes (if any) and to the output nodes and never backwards. Consequently, there are no cycles or loops in the network. Also a feed forward network treats all input features as unique and independent of one another. That is, a feed forward network has no notion of order in time, and the only input it considers is the current example it has been exposed to.

2.3 Convolutional Neural Network

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

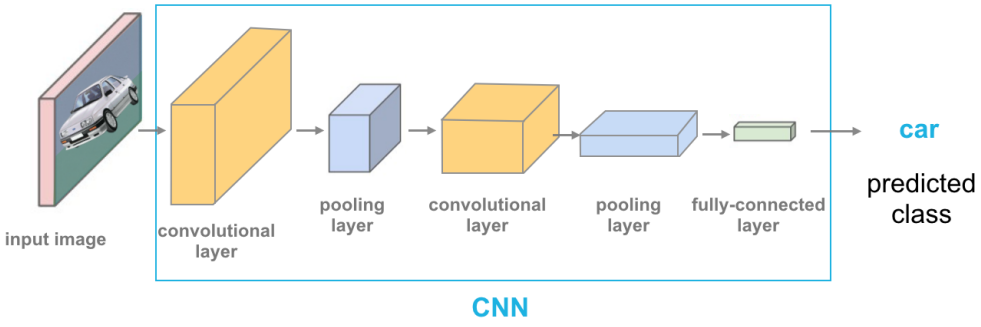

Figure 2.2: CNN

CNN consists of several operations, firstly a convolution operation (with which application of activation function is also associated) and then pooling operation. After doing several convolution and pooling operations the output of the last pooling layer is flattened (one by one each row of the matrix is kept into a vector) which is then fed into a fully connected feed forward neural network then the output of the neural network is processed as per the underlying problem.

Chapter 3

Problem Addressed and Methodology

3.1 Problem Addressed

The underlying problem of the project is weakly supervised object detection. In this the training of the deep learning based model happens with only image level labels (i.e. only names of object present in the image not their location) and we want labels as well as location of objects at test time. Fig: 3.1 shows the difference between training data for (Fully) supervised and weakly supervised learning.

(a) Supervised learning (b) Weakly supervised learning

Figure 3.1: Training instances for weakly and fully supervised object detectors [3]

3.2 Method

The overall idea for making weakly supervised deep detection network (WSDDN) consists of three steps. First, a CNN pre-trained on a large-scale image classification task (section 3.2.1) was obtained. Second, the WSDDN was constructed as an architectural modification of this CNN (section 3.2.2). Third, the WSDDN was trained/fine-tuned on a target dataset, once more using only image-level annotations (section 3.2.3). The remainder of this section discusses these three steps in detail.

3.2.1 Pre-trained network

The WSDDN is built on a pre-trained CNN that has been pre-trained on the ImageNet ILSVRC 2012 data [4] with only image-level supervision (i.e. no bounding box annotations). One of the pre-trained CNN used is VGG16 [5] whose architecture is shown in Fig: 3.2. More details of the used CNN architectures for experiments are given in section 4.

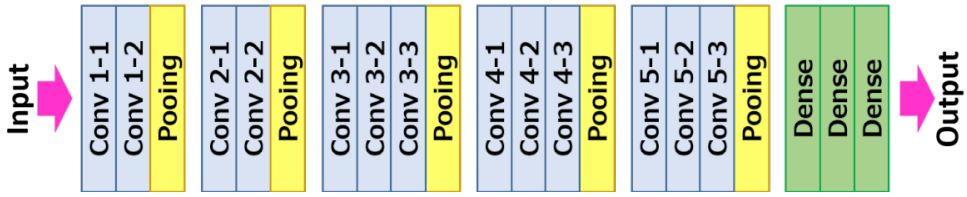

Figure 3.2: Architecture of VGG16

3.2.2 Weakly supervised deep detection network

Three modifications were introduced in the architecture of CNN obtained in last step (refer fig: 3.2):

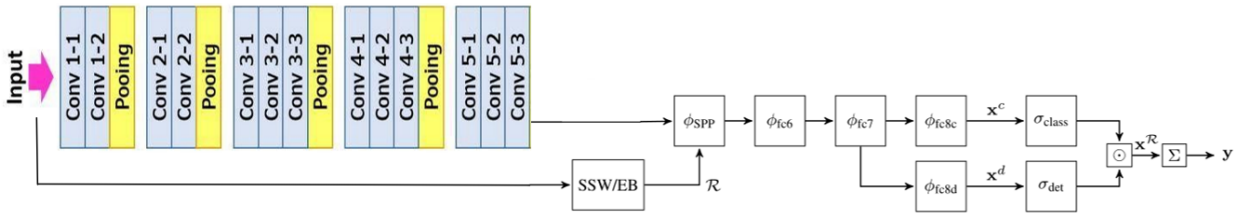

Figure 3.3: WSDDN: As a Modification of VGG16

1. Replaced the pooling layer in the last convolution block with a layer implementing spatial pyramid pooling (SPP) Layer [6].

   This SPP layer takes as input the feature map from last convolution layer and a set of region proposals (bounding boxes) and produces for each region a feature vector. These region proposals are obtained by region proposal mechanism such as Selective Search Window (SSW) [7] and Edge Box (EB) method [8].

   After this region-level features are further processed by two fully connected layers ϕ_{fc6} and ϕ_{fc7} then the flow is branched off into two data streams, described next.

2. A parallel detection branch was added to the classification one that contains a fully-connected layer followed by a soft-max layer.

Classification data stream: The first data stream performs classification of the individual regions, by mapping each of them to a C-dimensional vector of class scores. For this purpose the output of ϕ_{fc8c} (i.e. $X^c \in \mathbb{R}^{C\times|R|}$ where C is number of classes and $|R|$ is number of proposed regions) is passed by the following softmax operator,

$$[\sigma_{class}(\mathbf{x}^c)]_{ij} = \frac{e^{x^c_{ij}}}{\sum_{k=1}^{C} e^{x^c_{kj}}}$$

Detection data stream: The second data stream performs detection of the individual classes, by giving each region a relative score. For this purpose the output of ϕ_{fc8d} (i.e. $X^d \in \mathbb{R}^{C\times|R|}$ where C is number of classes and $|R|$ is number of proposed regions) is passed by the following softmax operator,

$$[\sigma_{det}(\mathbf{x}^d)]_{ij} = \frac{e^{x^d_{ij}}}{\sum_{k=1}^{|R|} e^{x^d_{ik}}}$$

3. Then the classification and detection streams are combined by element-wise product. The final score of each region is obtained by taking the element-wise product of output of both streams giving again a $C \times |R|$ size matrix. Then for each classes regions are ranked based on the final score followed by non-maxima suppression (i.e iteratively remove regions with intersection over union (IoU) more than 0.4 with already selected regions) giving detection for each classes.

   As the aim was to make a weakly supervised method i.e the model should use only image level labels (name of the objects in the images). So, an **Image level classification score** y_c as output of network was obtained by adding the score for each regions (adding the columns). Note that y_c is a sum of element-wise product of softmax normalised scores over $|R|$ regions and thus it is in the range of $(0, 1)$ and y is a vector with dimension as total number of classes.

$$y_c = \sum_{r=1}^{|R|} e^{x^R_{cr}}$$

3.2.3 Training WSDDN

For each training image x_i we have image level labels $y_i \in \{-1, 1\}^C$ and lets denote the final obtained output from network as $\phi^y(x_i|w)$, w is the set of all parameters of the CNN. Then stochastic gradient descent was performed to minimize the following Loss function w.r.t parameters w. In this loss function the loss for one sample can be viewed as sum of C binary-log-loss terms. As $\phi^y(x_i|w)$ is in range of (0, 1), it can be considered as a probability of class k being present in image x_i, i.e. $P(y_{ki} = 1)$. When the ground-truth label is positive, the binary log loss becomes $log(p(y_{ki} = 1))$ and $log(1 - p(yki = 1))$ otherwise.

$$E(\mathbf{w}) = \frac{\lambda}{2}||\mathbf{w}||^2 + \sum_{i=1}^{n}\sum_{k=1}^{C} log(y_{ki}(\phi^y_k(x_i|\mathbf{w}) - \frac{1}{2}) + \frac{1}{2})$$

Chapter 4

Experimentation and Results

4.1 Dataset and Performance metrics

I have performed the experiments on the PASCAL VOC 2007 [9], as it is the most widely-used benchmark in weakly supervised object detection. The VOC 2007 dataset consists of 2501 training, 2510 validation, and 5011 test images containing bounding box annotations for 20 object categories. The training and validation splits both are used for training and reported results evaluated on test split.

For detection, standard PASCAL VOC protocol is used as performance measures. It reports average precision (AP) at 50% intersection-over-union (IoU) of the detected boxes with the ground truth ones.

4.2 Experiments and Results

Three networks (WSDDNs) are formed using different base models. The first network is based on the AlexNet [10]. I will refer to this network as A. The second one is based on VGG16 [5] which has more depth, it uses small receptive fields and smaller stride length as compared to AlexNet. I will refer to this network as B. The last network is based on the deep VGG19 model [5] this is almost same as VGG16 but with more layers. I will refer to this network as C. As described before these models are pre-trained on the ImageNet ILSVRC 2012 challenge data [4]

The WSDDNs are trained on the PASCAL VOC training and validation data by using fine-tuning on all layers. Here, fine tuning performs the essential function of learning the classification and detection streams, effectively causing the network to learn to detect objects, but using only weak image-level supervision. The experiments are run for 20 epochs and the learning rate for different epochs are given in table:4.1

Table 4.1: learning rate for non-regularized

Model ↓ *Epochs* →	0 − 10	10-20
A	10^{-5}	10^{-6}
B	50^{-6}	10^{-6}
C	50^{-6}	50^{-7}

Table 4.2: learning rate for regularized model

Model ↓ *Epochs* →	0-10	10-20
A	10^{-5}	10^{-6}
B	50^{-6}	10^{-6}
C	10^{-6}	50^{-7}

Figure 4.1: Detection results

In order to generate candidate regions to use with the networks, I used two proposal methods, Selective Search Windows (SSW) [7] for detection in images for which pre-computed region proposals are not present (using this setting some detection result is shown in fig 4.1), and EdgeBoxes (EB) [8] for training and testing on test set. In addition to region proposals, EB provides an objectness score for each region based on the number of contours (i.e. closed edge curves) wholly encloses [see fig 4.2]. This additional information is exploited by multiplying the feature map ϕ_{spp} proportional to its score via a scaling layer in WSDDN and denote this setting as Box Sc. Since the networks uses a SPP layer to aggregate descriptors for each region, images do not need to be resized to a particular size as in the original pre-trained model. Instead, the original aspect ratio of images are kept fixed and resized to five different scales (setting their maximum of width or height to 480, 576, 688, 864, 1200 respectively). During training, random horizontal flips are applied to the images and selected at a scale at random as a form of data augmentation. At test time the average of the outputs of 10 images (i.e. the 5 scales and their flips) is taken.

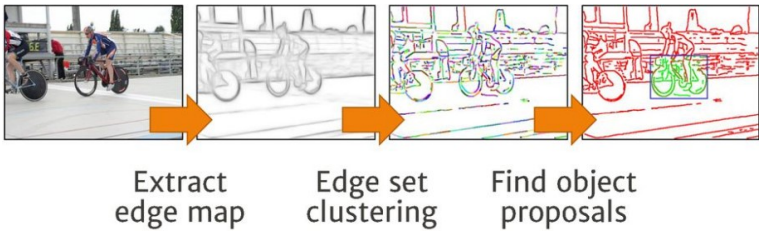

Figure 4.2: Edge Box method

I have also applied a soft regularisation strategy, **Spatial Regularization**, that penalises the feature map discrepancies between the highest scoring region and the regions with at least 60% IoU during training. The regularizer is defined below which is then added to the loss function described earlier.

$$\frac{1}{nC}\sum_{k=1}^{C}\sum_{i=1}^{n_k^+}\frac{1}{2}\phi_k^y(x_i|w)(\phi_{kp}^{fc7}-\phi_{ki}^{fc7})^T(\phi_{kp}^{fc7}-\phi_{ki}^{fc7})$$

where n_k^+ is the number of positive images for class k and $kp = argmax_j \phi_{kj}^y$ is the highest scoring region in image i for class k. Applying this regularization and training the models with learning rate as given in table 4.2 helped me to get better performance along with training for fewer epochs. The plot of training loss for un-regularised models is shown in fig 4.3 and for regularised models is shown in fig 4.4. The plots shows that introducing regulariser made the loss function smooth i.e. the optimization process is more smoother in the regularised models.

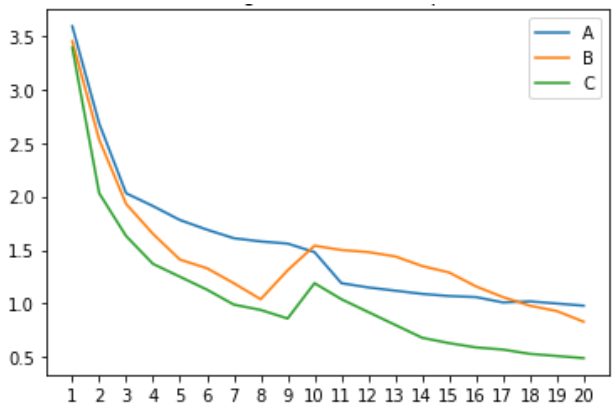

Figure 4.3: Training loss for un-regularized model

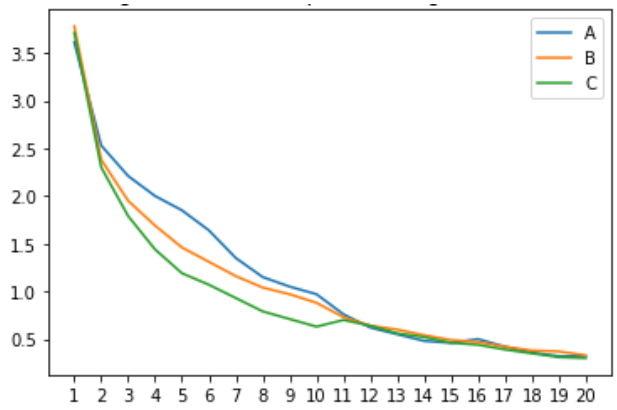

Figure 4.4: Training loss for regularized model

The time taken for training one epoch for model A was 15 minutes and testing time was 2 hours. Whereas for model B and C training time for one epoch was 45 minutes and testing time was 2 hours 30 minutes. Given the fact that AlexNet, VGG16 and VGG19 has 8, 16 and 19 learnable layers and 62M, 132M and 143M parameters respectively. So, it can seen that this difference in time is associated with the difference in the depth (complexity) of the base models. (Time given here are approx.)

The results obtained by me are provided in Table 4.3 which is comparable to the results, in Table 4.4, given in the original paper. The authors of original paper [1] had used three base models, which are almost same as of mine, and named their networks as S, M and L. (for making ensemble I have used publicly available Python library 'ensemble-boxes' [11])

In order to reduce the time required for training and testing I have also experimented by

Table 4.3: my results

	A	B	C	ens
EB+Box score	26.7	28.2	30.6	32.6
EB+Box sc.+sp. reg	27.8	32.1	32.2	34.7

Table 4.4: results in original paper

	S	M	L	ens
EB+Box score	33.4	32.7	30.4	36.7
EB+Box sc.+sp. reg	34.5	34.9	34.8	39.3

Table 4.5: VOC 2007 test detection mean average precision (%). The ensemble network is denoted as Ens.

Figure 4.5: Classification results

selecting top 50% boxes based on EB score for training and testing which reduced the respective time taken by a factor of 2 with a drop of around only 2-3% in the performance.

I have also calculated CorLoc (with criterion IoU 0.2) on the PASCAL VOC 2007 positive test image and got 44.0%, 51.6% and 50.8% by model A, B and C respectively and 58.8% with the ensemble model. I have tested all the models on PASCAL VOC classification task also and have got 81.7%, 89.8% and 89.9% Average Precision (AP) by A, B and C respectively. In contrast the max AP reported in the original paper is 89.7%. Some sample results from classification task is shown in fig 5.1

We note that pre-computed region proposals by EB method are used for training and testing. In order to make it work on custom images (images without pre-computed regions) I have used selective search window method of region proposal and then based on scores obtained by the model the detection is done. The detection on a image was taking around 2 secs with around 50 region proposals by selective search window. The average number of pre-computed region(Bounding Box) proposal by EB method was around 4000. Hence detection without pre-computed region proposals does not perform at par because very few region proposals are highly likely to miss the region of interest. So, there is a trade-off between better detection and time required when doing detection using these models.

Chapter 5

Conclusion and Future Work

In this project I have trained and experimented with the network proposed by Bilen et. al. [1]. I have used AlexNet, VGG16 and VGG19 to make three different models for the purpose of weakly supervised object detection and obtained results comparable to those in paper. I was able to reduce the training and testing time by a factor of 2 by selecting only top 50% based on the EB scores of the proposed region by EB method. Apart from object detection I have also tested my models on the task of image classification on PASCAL VOC test data and got my best AP as 0.2% more than the best AP of paper.

In case of false detection two types of error were dominant (1) It grouped multiple object instances with a single bounding box (2) Detecting most discriminating parts (e.g. "faces") rather than whole object.

Figure 5.1: Major mistakes

I used this model for detection on random images but it was taking around 2 sec for an image most of this time was taken by region proposal. Hence, this can be improved in future by using more efficient region proposals.

As mentioned the testing was done on test data by taking average of scores obtained for 10 different augmented images of each image in test data. Instead of just average, in future I will try taking ensemble of the detections from all 10 augmentations of each image to make final detection for the each image and hopefully it will improve the overall performance.

Bibliography

[1] Bilen, Hakan, and Andrea Vedaldi. "Weakly supervised deep detection networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[2] Image by Sabrina Jiang © Investopedia 2020.

[3] Jeong, Jisoo et al. "Consistency-based Semi-supervised Learning for Object detection." Neural Information Processing Systems (2019).

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. IJCV, pages 1–42, April 2015.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", Proc. Int. Conf. Learn. Represent., 2015.

[6] Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In CVPR, pages 2169–2178, 2006.

[7] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. IJCV, 104(2):154–171, 2013.

[8] C. L. Zitnick and P. Doll´ar. Edge boxes: Locating object proposals from edges. In ECCV, pages 391–405. 2014.

[9] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, pages 1097–1105. 2012.

[11] Solovyev, Roman and Wang, Weimin and Gabruseva, Tatiana "Weighted boxes fusion: Ensembling boxes from different object detection models", Image and Vision Computing, page 1-6, year 2021, publisher Elsevier