Deep Learning Models for Context-Aware Object Detection

Seyyed Mostafa Arefiyan Khalilabad

Thesis submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science in Computer Engineering

A. Lynn Abbott, Chair
Naren Ramakrishnan
Pratap Tokekar

August 11, 2017
Blacksburg, Virginia

Keywords: Context-aware Detection, Object Detection, Context Modeling, Context Extraction, Convolutional Neural Network, Computer Vision, Deep Learning, Machine Learning
Copyright 2017, Mostafa Arefiyan
Deep Learning Models for Context-Aware Object Detection

Seyyed Mostafa Arefiyann Khalilabad

(ABSTRACT)

In this thesis, we present ContextNet, a novel general object detection framework for incorporating context cues into a detection pipeline. Current deep learning methods for object detection exploit state-of-the-art image recognition networks for classifying the given region-of-interest (ROI) to predefined classes and regressing a bounding-box around it without using any information about the corresponding scene. ContextNet is based on an intuitive idea of having cues about the general scene (e.g., kitchen and library), and changes the priors about presence/absence of some object classes. We provide a general means for integrating this notion in the decision process about the given ROI by using a pretrained network on the scene recognition datasets in parallel to a pretrained network for extracting object-level features for the corresponding ROI. Using comprehensive experiments on the PASCAL VOC 2007[7], we demonstrate the effectiveness of our design choices, the resulting system outperforms the baseline in most object classes, and reaches 57.5 mAP (mean Average Precision) on the PASCAL VOC 2007 test set in comparison with 55.6 mAP for the baseline.
Deep Learning Models for Context-Aware Object Detection

Seyyed Mostafa Arefiyan Khalilabad

(GENERAL AUDIENCE ABSTRACT)

The object detection problem is to find objects of interest in a given image and draw boxes around them with object labels. With the emergence of deep learning in recent years, current object detection methods use deep learning technologies. The detection process is solely based on features which are extracted from several thousand regions in the given image. We propose a novel framework for incorporating scene information in the detection process. For example, if we know the image is taken from a kitchen, the probability of seeing a cow or an airplane decreases and observation probability of plates and persons increases. Our new detection network uses this intuition to improve the detection accuracy. Using extensive experiments, we show the proposed methods outperform the baseline for almost all object types.
## Contents

1 Introduction .................................................. 1
   1.1 Generic Deep Object Detection Pipeline ................. 1
   1.2 Motivation for Context .................................. 2

2 Related Work .................................................. 5

3 Context Modeling ............................................. 9
   3.1 Explicit Context Modeling ................................. 9
   3.2 Implicit Context Modeling ............................... 10

4 Network Architecture ....................................... 12
   4.1 Explicit Context Network ................................. 12
       4.1.1 Early Fusion Architecture .......................... 12
       4.1.2 Late Fusion Architecture ........................... 13
   4.2 Implicit Context Network ................................. 14

5 Experiments and Results ..................................... 16
   5.1 Dataset Overview .......................................... 16
   5.2 Experiments Setup ......................................... 17
   5.3 Results ..................................................... 17
   5.4 Ablation studies .......................................... 24
       5.4.1 Explicit vs. implicit context networks ............ 24
       5.4.2 Effect of explicit context signal ................. 24
5.4.3 Effect of fine-tuning ........................................ 24
5.4.4 Scene classification vs. scene embedding .............. 25
5.4.5 Scene classification vs. scene-object classification .... 25

6 Future Work .................................................. 27

7 Conclusion ....................................................... 28

Bibliography ..................................................... 29

Appendix A More Results’ Analysis .......................... 33
A.1 Detection Types ............................................. 33
A.2 Detection Trends .......................................... 36
## List of Figures

1.1 Generic Deep Object Detection Pipeline. A generic object detection pipeline consists of the following steps (for simplicity ignoring the pre/post-processing steps): (1) Extracting features from the given image using a pretrained convolutional neural network (CNN). (2) Proposing some regions (region-of-interest, i.e., ROI) in the image for further consideration as potential objects. The regions can be produced by an object proposal method such as Edge Boxes [36] or Selective Search [31]. (3) Extracting features corresponding to the given ROI from the feature map. (4) Classifying the ROI as one of object classes or background using a classifier module and predicting a bounding-box for the given instance of object classes by a bounding-box regressor (Initial photo courtesy of Everingham et al. [7]).

1.2 Motivation. In most cases, overall information about the entire image helps to make assumptions about the existence/absence of some object types. This information can be utilized to encourage/suppress detections of the object classes. If we know the image above has been taken from a farm, object classes related to the farms such as tree are more likely to be presented in the image than the ones unrelated such as zebra (Initial photo courtesy of Blevins [6]).

2.1 DPM with Context [22]. The colored boxes around the root filter ($p_0$) show context cues for the top, left, bottom, and right of the object. The white box ($p_i$) represents object parts (Initial photo courtesy of Lulko [21]).

2.2 R-CNN Pipeline (Figure courtesy of Girshick et al. [11]).

2.3 Fast-RCNN Pipeline [10]. The image is fed to the convolutional layers of a pretrained AlexNet to produce the feature map. An ROI-pooling layer maps the given ROI to the corresponding region in the feature map and generates a fixed-length vector. The pretrained fully connected layers (FCs) from AlexNet are used to generate a 4096-length vector. Two other FCs are used to classify the ROI and create a bounding-box (Figure courtesy of Girshick [10]).
2.4 Inside-Outside Net (ION) Pipeline \cite{5}. Convolutional features are extracted from different layers in VGG16 \cite{29}. Two 4-direction RNN extract features for each location which represent spatial dependencies. These features are normalized and concatenated for each ROI for classification and regression a bounding-box (Figure courtesy of Bell et al.\cite{3}).

2.5 Multi Region CNN Pipeline (Figure courtesy of Gidaris et al.\cite{9}).

2.6 ParseNet Pipeline (Figure courtesy of Liu et al.\cite{18}).

3.1 Explicit Context Modeling. The explicit context modeling approach has two parallel pipelines: one for extracting object features from the image and another one for extracting context features from the entire image, i.e., context extraction module (light-green parts). This module can be another CNN which has been trained for a scene recognition task such as MIT Place dataset \cite{35,34}. It extracts features from the entire image and concatenates them to the features for each ROI which has been extracted by a regular CNN (trained on ImageNet \cite{27}). The rest of the pipeline use these augmented features for classifying and regressing a bounding-box (Initial photo courtesy of Everingham et al.\cite{7}).

3.2 Implicit Context Modeling. The implicit context modeling approach uses a genetic object detection pipeline (see Section 1.1) but the difference is the pretrained CNN is capable of recognizing scenes as well as objects. The CNN has been trained on the combination of MIT Places dataset \cite{35} and ImageNet \cite{27}. Therefore in this way, the pipeline implicitly takes the context into the account for detecting objects (Initial photo courtesy of Zhou et al.\cite{33}).

4.1 Early Fusion Explicit Context Architecture. The architecture consists of two parallel networks for extracting object and context features. It is an extension of Fast-RCNN \cite{10}. The first branch extracts convolutional features from the given images. This CNN has been pretrained on ImageNet dataset. An ROI pooling layer provides features related to the ROI of interest. The second network provides context features for the ROI. It has been pretrained on MIT Places dataset. It takes the entire image, warps it to the size of network input and extract features from the last layers. Then a scene-pooling layer concatenates the features from ROI-pooling layer with these context features. The architecture has two other fully-connected layers and then a classifier predicts the class for the given ROI, and a bounding-box regressor refines its bounding-box location (Initial figure courtesy of Girshick\cite{10} and Zhou et al.\cite{33}).
4.2 Late Fusion Explicit Context Architecture. This architecture is very similar to the early fusion architecture. The only difference is the scene-pooling layer concatenates the context feature with object feature in the last stage of the pipeline just before the classifier and regressor (Initial figure courtesy of Girshick [10] and Zhou et al. [33]).

4.3 Implicit Context Architecture. This architecture is almost identical to the Fast-RCNN [11], the only difference is in the Fast-RCNN pipeline, they use a pretrained network on ImageNet dataset but here we use a network which has been pretrained on both ImageNet and MIT Places datasets. In this way, our pipeline can use context cues in addition to object features (Initial figure courtesy of Girshick [10] and Zhou et al. [33]).

5.1 The PASCAL VOC 2007 dataset (Figure courtesy of Everingham et al. [7]).

5.2 Detection type statistics for different sets of classes using [12]. Each plot shows the percentage of each detection type for each network and object category set. Detection types are as follows: Correct detection (white), Localization -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Similar -false positive due to confusion with similar object classes- (red), Other -false positive due to confusion with dissimilar object classes- (green), Background -false positive due to confusion with background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively. The late-fusion architecture increases the detection accuracies in the most cases.

5.3 Distribution of detection type for Furniture classes using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
5.4 Distribution of detection type for Vehicle classes using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

5.5 Distribution of detection type for Animal classes using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

A.1 Detection type statistics for Aeroplane class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
A.2 Detection type statistics for Boat class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X EF, and ContextNet-X + LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

A.3 Detection type statistics for Car class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X + LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

A.4 Detection type statistics for Person class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X + LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

A.5 Detection type statistics for TV/Monitor class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X + LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
A.6 Distribution of detection type for Aeroplane class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

A.7 Distribution of detection type for Boat class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

A.8 Distribution of detection type for Car class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
A.9 Distribution of detection type for Person class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

A.10 Distribution of detection type for TV/Monitor class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Results on the PASCAL VOC 2007 (precision reported). All networks have been trained on the PASCAL VOC 2007 trainval set. ContextNet-I is an implicit context network while ContextNet-X EF and ContextNet-X+ LF are the explicit ones. ContextNet-X EF/ContextNet-X+ LF utilizes an/a early/late-fusion architecture. ContextNet-X+ uses an AlexNet which has been trained on both ImageNet and MIT places datasets as the object feature extractor.</td>
<td>19</td>
</tr>
<tr>
<td>5.2</td>
<td>Implicit &amp; Explicit Context Networks precision Comparison on the PASCAL VOC 2007 test set. ContextNet-I/ContextNet-X is an implicit/explicit context network. ContextNet-X exploits an early fusion architecture. Both networks have been trained on the PASCAL VOC trainval set.</td>
<td>19</td>
</tr>
<tr>
<td>5.3</td>
<td>Explicit context cues effect on models’ precision on the PASCAL VOC 2007 test set. ContextNet-I and ContextNet-X+ are an implicit and a late-fusion explicit context network, respectively. Both networks use an AlexNet for object feature extraction which has been trained on both ImageNet and MIT Places datasets. ContextNet-X+ benefits from a context extraction module. Both networks have been trained on the PASCAL VOC trainval set.</td>
<td>19</td>
</tr>
<tr>
<td>5.4</td>
<td>Fine-tuning effect on models’ precision on the PASCAL VOC 2007 test set. ContextNet-X+/ContextNet-X+ FT is a late-fusion explicit context network. Both networks use an AlexNet for object feature extraction which has been trained on both ImageNet and MIT Places datasets. In ContextNet-X+, the context extraction part has been frozen but in ContextNet-X+ FT is fine-tuned on the PASCAL VOC 2007 trainval set.</td>
<td>26</td>
</tr>
<tr>
<td>5.5</td>
<td>Scene classification &amp; scene embedding precision comparison on the PASCAL VOC 2007 test set. ContextNet-X+ Em/ContextNet-X+ is an explicit context network. Both networks have been trained on the PASCAL VOC trainval set and use the late-fusion architecture. ContextNet-X+ Em uses the FC7 layer of AlexNet as context cues instead of FC8.</td>
<td>26</td>
</tr>
</tbody>
</table>
Scene & Scene-Object classification precision comparison on the PASCAL VOC 2007 test set. ContextNet-X$^+$ Hyb/ContextNet-X$^+$ is an explicit context network. Both networks have been trained on the PASCAL VOC trainval set and use the late-fusion architecture. ContextNet-X$^+$ Hyb uses an AlexNet which has been trained on both ImageNet and MIT Places dataset as the context extraction module.
Chapter 1

Introduction

Computer vision problems and techniques have gained significant interest in recent years. Part of this interest is because of tens of million images and videos, which users upload daily on social networks such as Facebook and Instagram and the needs of understanding and extracting knowledge from these valuable data. Another aspect is moving from toy datasets such as Caltech101 to much bigger datasets such as ImageNet, MIT Places, MS-COCO, and Youtube-8M. The availability of these data enables us to train models with several hundred million parameters, which was unimaginable in the past.

As part of these advances, the object detection field has been transformed from using hand-crafted features and shallow models to the use of very deep convolutional neural networks (CNNs) which automatically learn features that are useful for the given task. These changes improve detection accuracies dramatically in comparison with the past shallow models. Despite these improvements, these deep models do not employ context cues in the detection process (e.g., the Faster-RCNN, SSD, and YOLO models). In this thesis, we introduce a novel framework for integrating context cues in a general object detection pipeline, and we report results on the PASCAL VOC 2007 dataset.

The rest of this chapter is as follows. First in Section 1.1, we explain a generic pipeline which uses deep learning techniques for object detection problem. This is a generalization of standard techniques in the field such as R-CNN, Fast-RCNN, Faster-RCNN, YOLO, and SSD models. Then we motivate the idea of using context in Section 1.2.

1.1 Generic Deep Object Detection Pipeline

In the deep learning era, a generic model for detecting objects in the given image is as follows (Figure 1.1). We have omitted the pre/post-processing steps in Figure 1.1 to keep the diagram simple. The input image is warped to a predefined size and a mean-value subtraction...
Figure 1.1: Generic Deep Object Detection Pipeline. A generic object detection pipeline consists of the following steps (for simplicity ignoring the pre/post-processing steps): (1) Extracting features from the given image using a pretrained convolutional neural network (CNN). (2) Proposing some regions (region-of-interest, i.e., ROI) in the image for further consideration as potential objects. The regions can be produced by an object proposal method such as Edge Boxes [36] or Selective Search [31]. (3) Extracting features corresponding to the given ROI from the feature map. (4) Classifying the ROI as one of object classes or background using a classifier module and predicting a bounding-box for the given instance of object classes by a bounding-box regressor (Initial photo courtesy of Everingham et al. [7]).

is done on each pixel. Hundreds or several thousand regions are determined as regions-of-interest (ROIs) which are candidates for object instances. These ROIs are generated using different approaches such as object proposal methods (e.g., Selective Search [31] and Edge Boxes [36]) or putting predefined grids with different scales and aspect-ratios on top of the image (e.g., SSD [17], YOLO [24, 25], and G-CNN [23]). Then the image is forwarded to the convolutional layers of a pretrained CNN to obtain a feature map for the entire image. An ROI-pooling layer extracts the corresponding part of the feature map related to the current ROI and interpolates it to fit the input size of the remaining part of the network. Subsequently, the classifier predicts a class label (object classes of interest or background) and the regressor provides a bounding-box around the object instance for the given ROI. And finally, a non-maximum suppression is performed over all predicted bounding-boxes in the given image to remove redundant detections of same object instances.

1.2 Motivation for Context

As we mentioned earlier, in the current deep learning approaches to object detection, no one has proposed a method for employing context or scene cues for detecting objects. Therefore, the previous methods may confuse the background, similar, or dissimilar classes with the object of interest. If we know in advance that the given image is from a specific type of scene
Figure 1.2: Motivation. In most cases, overall information about the entire image helps to make assumptions about the existence/absence of some object types. This information can be utilized to encourage/suppress detections of the object classes. If we know the image above has been taken from a farm, object classes related to the farms such as tree are more likely to be presented in the image than the ones unrelated such as zebra (Initial photo courtesy of Blevins[6]).

(e.g., farm), without looking at the image, we can imagine what types of objects are most likely to be presented (e.g., horse, person, and cow), and what object classes are unlikely to observed (e.g., airplane and zebra) (Figure 1.2).

With this observation, we can use methods which are able to recognize scenes and incorporate their scene classification output to the object detection pipeline. Therefore, when the detector wants to predict an object class, it can incorporate the scene/context information into its decision, and encourage detection of classes which are compatible with the scene and suppress detections which are not related to the scene.

From Bayesian perspective, if we have object class $C$, data from bounding box $B$, and scene $S$, assuming data from bounding box $B$ and scene $S$ are conditionally independent$^1$, we can derive the following:

$^1$We assume the probability of observing scene $S$ and data from bounding box $B$ (i.e., pixels’ values) given object class $C$ (i.e., $P(S, B|C)$) are conditionally independent. It means if we know the object is a horse, given knowledge that the horse is black, does not have any influence on probability of occurrence of the scene (e.g., farm or meadow) and given the scene is farm does not provide any information about the color of horse.
From (1.1) to (1.4), the following can be entailed:

\[ P(C|S, B) \propto P(B, S|C)P(C) \quad (1.1) \]
\[ \propto P(B|C)P(S|C)P(C) \quad (1.2) \]
\[ \propto P(C|B)P(S|C) \quad (1.3) \]
\[ \propto P(C|B)P(C|S)P(S) \quad (1.4) \]

From (1.1) to (1.4), the following can be entailed:

\[ P(C|S, B) \propto P(C|B)P(C|S)P(S) \quad (1.5) \]

If we consider a uniform distribution for the scene prior (i.e., \( P(S) \)), we reach the following equation:

\[ P(C|S, B) \propto P(C|B)P(C|S) \quad (1.6) \]

For the farm example, using (1.6), if we have the following, \( P(C = Zebra|B = PurpleBox) = 0.9, \ P(C = Horse|B = PurpleBox) = 0.4, \ P(C = Zebra|S = Farm) = 0.05, \) and \( P(C = Horse|S = Farm) = 0.5 \), we reach these probabilities for zebra and horse classes:

\[ P(C = Zebra|S = Farm, B = PurpleBox) \propto \]
\[ = 0.9 \times 0.05 \]
\[ = 0.045 \quad (1.7) \]

\[ P(C = Horse|S = Farm, B = PurpleBox) \propto \]
\[ = 0.4 \times 0.5 \]
\[ = 0.20 \quad (1.8) \]

As you can see in (1.7), the probability of detecting a zebra has decreased a lot and the prediction for horse class has encouraged in (1.8) because it is more compatible with a farm scene.
Chapter 2

Related Work

Using context for better object detection and segmentation has been studied for a long time\cite{14, 4, 22, 11, 9, 2, 5, 18, 28, 15}. Different approaches and levels of abstraction have been proposed to model context, from local cues/patches in the images to general information about the images as a whole. The more relevant approaches have been described briefly in the following.

Mottaghi et al.\cite{22} propose a deformable part model (DPM) which explicitly models the context using global and local parts (Figure 2.1). The global context, in their model, refers to other object classes which can be present or absent in the entire image. The contextual classes around the object of interest are considered as the local context.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{dpm_context.png}
\caption{DPM with Context\cite{22}. The colored boxes around the root filter ($p_0$) show context cues for the top, left, bottom, and right of the object. The white box ($p_i$) represents object parts (Initial photo courtesy of Lulko\cite{21}).}
\end{figure}

Girshick et al.\cite{11} extract 2k object proposals using selective search method\cite{31}, then extend each of them with a 16-pixel margin as context. The features for each object proposal is extracted by employing a CNN which is pretrained in the ImageNet dataset\cite{27}. In the final stage, a support vector machine classifies each object proposal and a regressor predicts the bounding box (Figure 2.2).
Girshick extends the previous work (Figure 2.3), known as fast-RCNN [10], to reduce the computational overhead of extracting convolutional features of overlapping object proposals repeatedly. In the Fast-RCNN pipeline, the convolutional features are computed from the entire image first, then each object proposal is mapped to the corresponding region of feature map using a region-of-interest (ROI) layer. The corresponding features are warped to fit the input size of the fully connected layer of the network in-hand. The remaining parts of the pipeline are same as the R-CNN.

Bell et al. [5] introduce Inside-Outside Net (ION) architecture (Figure 2.4) which consists of two 4-direction recurrent neural networks (RNNs) after the convolutional layer 5 in VGG16 [29]. These two RNNs capture the short-term and long-term spatial dependencies between features in the image and represent the context for each location in the image. For each object proposal, the convolutional features from different layers in the VGG16 and these contextual features are concatenated and the classification and bounding-box regression are performed on top of these features.
Figure 2.4: Inside-Outside Net (ION) Pipeline [5]. Convolutional features are extracted from different layers in VGG16 [29]. Two 4-direction RNN extract features for each location which represent spatial dependencies. These features are normalized and concatenated for each ROI for classification and regression a bounding-box (Figure courtesy of Bell et al. [5]).

Gidaris et al. [9] introduce a multi-region CNN architecture (Figure 2.5) which uses a set of region adaptation modules after the convolutional layers in the CNN. Each region adaptation module is enforced to detect a specific region of the original or extended ground truth bounding boxes. The extended ground truth bounding boxes are responsible for modeling context in this architecture.

Liu et al. [18] incorporate global context to a fully convolutional network [19] for improving its semantic segmentation performance. They use the features’ average value from the last fully-connected layer as the global context, called it global average pooling layer, and concatenate them to the last feature map for segmenting the image. These global average features reduce
the confusion between classes which are not compatible with the entire image.

Figure 2.6: ParseNet Pipeline (Figure courtesy of Liu et al.[18]).

Azizpour et al.[2] suggest a general discriminative latent variable framework which benefits from both positive and negative latent variables for prediction. The negative latent variables play the counter-evidence role in their framework, i.e., if they are detected the classifier confuses the object with the similar classes. From object detection perspective, the positive and negative latent variables can be used for modeling context, objects, and object parts in the image.
Chapter 3

Context Modeling

In this chapter, we introduce different approaches for modeling and incorporating context cues in the object detection pipelines. As we mentioned in chapter [1], we consider context as a global signal which describes the entire scene, such as labels like library, airport, and kitchen. This definition is compatible with scene recognition datasets such as the MIT Places [35], MIT Places2 [34], and SUN [32]. We can exploit this property and propose two different approaches as follows: explicit context modeling and implicit context modeling.

3.1 Explicit Context Modeling

In this approach, we explicitly model context using features or signals which give us evidence about the entire image and describe the entire image as a whole (Figure 3.1). This pipeline has a dedicated module for modeling context and consists of two parallel parts:

- Context Feature Extraction: This part is a CNN which is pretrained on a scene recognition dataset. We use the MIT Places dataset [35] because their labels for the scenes are compatible with our definition of context. In addition, it contains a diverse set of scenes (205 scene categories) and a huge number of images (more than 2.4 million). These properties enable us to have a good scene feature extractor. There are two ways to model context using this pretrained CNN, either using scene classification probabilities or outputs of an intermediate layer. We conduct experiments and report results for both ways in Chapter [5].

- Object Feature Extraction: This module is another CNN which is pretrained on an object recognition task, using the ImageNet [27]. It operates over the entire image and produces a feature map. Then an ROI-pooling layer extracts feature corresponding to the given object candidate (i.e., ROI) in the image from this feature map. Because
these features are solely based on the given object candidate and extracted via an object-recognition CNN, we call them object features.

Finally, we concatenate object and context features, classify the given ROI and regress a bounding-box on these augmented object features.

Figure 3.1: Explicit Context Modeling. The explicit context modeling approach has two parallel pipelines: one for extracting object features from the image and another one for extracting context features from the entire image, i.e., context extraction module (light-green parts). This module can be another CNN which has been trained for a scene recognition task such as MIT Place dataset \[35, 34\]. It extracts features from the entire image and concatenates them to the features for each ROI which has been extracted by a regular CNN (trained on ImageNet \[27\]). The rest of the pipeline use these augmented features for classifying and regressing a bounding-box (Initial photo courtesy of Everingham et al.\[7\]).

3.2 Implicit Context Modeling

In addition to the explicit modeling of context, we can model the context in an implicit way. In this approach, we do not have a dedicated module for extracting context features but the idea is the object feature extractor of the previous section (Section 3.1) be able to detect and extract context features also.

For reaching this goal, we can train a network on an extended dataset which consists of both ImageNet and MIT Places datasets (Figure 3.2). The trained network learns useful features
for extracting knowledge for object recognition as well as scene classification. Therefore, by replacing the base network of generic pipeline in Section 1.1 with this pretrained CNN, the resulted pipeline utilizes context information and object features for detections, so it should outperform a network which is pretrained on just the ImageNet dataset.

Figure 3.2: Implicit Context Modeling. The implicit context modeling approach uses a genetic object detection pipeline (see Section 1.1) but the difference is the pretrained CNN is capable of recognizing scenes as well as objects. The CNN has been trained on the combination of MIT Places dataset[35] and ImageNet[27]. Therefore in this way, the pipeline implicitly takes the context into the account for detecting objects (Initial photo courtesy of Zhou et al.[33]).
Chapter 4

Network Architecture

As we mentioned in Chapter 2, there are a lot of efforts for improving object detection algorithms by employing context cues. In this chapter, we explain three novel architectures for modeling context in an object detection framework. The idea is general and can be applied in any object detection frameworks, but for illustration purposes, we use the Fast-RCNN framework\cite{10}. In Section 4.1 two architectures are proposed which introduce explicit context cues to the pipeline and in Section 4.2 an implicit approach for using context is suggested.

4.1 Explicit Context Network

In the explicit context modeling approach (see Chapter 3), we have a dedicated module for extracting context features from the given image and combining them with object features which are extracted from the given region-of-interest (ROI). This fusion can be done in early/late stages in the pipeline. In Subsection 4.1.1 we propose an early fusion architecture which follows by a late fusion architecture in Subsection 4.1.2.

4.1.1 Early Fusion Architecture

For early fusion architecture, we change the Fast-RCNN pipeline for incorporating the context features (see Figure 4.1). As we saw in Chapter 2, the Fast-RCNN pipeline uses a pretrained AlexNet for object detection. The convolutional part of AlexNet (i.e., conv1 to conv5 layers), provides the convolutional features for an ROI-pooling layer which crops and warps the given area in the feature map to the input size of the FC6 layer (a 4096-dimensional vector). In parallel, we use another AlexNet network which has been pretrained on MIT Places1 dataset. The input image is warped to fit the input size of the network.
and the network’s predictions are concatenated with the ROI features and fed into the fully-connected layers to produce a 4096-dimensional vector as input for the classifier and bounding-box regressor.

Figure 4.1: Early Fusion Explicit Context Architecture. The architecture consists of two parallel networks for extracting object and context features. It is an extension of Fast-RCNN[10]. The first branch extracts convolutional features from the given images. This CNN has been pretrained on ImageNet dataset. An ROI pooling layer provides features related to the ROI of interest. The second network provides context features for the ROI. It has been pretrained on MIT Places dataset. It takes the entire image, warps it to the size of network input and extract features from the last layers. Then a scene-pooling layer concatenates the features from ROI-pooling layer with these context features. The architecture has two other fully-connected layers and then a classifier predicts the class for the given ROI, and a bounding-box regressor refines its bounding-box location (Initial figure courtesy of Girshick[10] and Zhou et al.[33]).

4.1.2 Late Fusion Architecture

The late fusion architecture is similar to the early fusion but merges the context features in the late stage of pipeline just before the classifier and bounding-box regressor (Figure 4.2). In this architecture, we can benefit from a pretrained FC6 layer of Alexnet as a fully-connected layer just after the ROI layer. In the early fusion, the input size of this layer is changed (see
Figure 4.2: Late Fusion Explicit Context Architecture. This architecture is very similar to the early fusion architecture. The only difference is the scene-pooling layer concatenates the context feature with object feature in the last stage of the pipeline just before the classifier and regressor (Initial figure courtesy of Girshick [10] and Zhou et al. [33]).

Figure 4.1), so we cannot use a pretrained layer and should train this from scratch.

4.2 Implicit Context Network

The implicit context network exploits the Fast-RCNN pipeline but utilizes a pretrained AlexNet network which is capable of recognizing scene/context in addition to objects (Figure 4.3). This is achieved by training AlexNet on both ImageNet and MIT Places dataset. Therefore the features provided by this network contain cues about the objects as well as context. For this architecture, we replace the layers with this context-aware AlexNet model.
Figure 4.3: Implicit Context Architecture. This architecture is almost identical to the Fast-RCNN\[11\], the only difference is in the Fast-RCNN pipeline, they use a pretrained network on ImageNet dataset but here we use a network which has been pretrained on both ImageNet and MIT Places datasets. In this way, our pipeline can use context cues in addition to object features (Initial figure courtesy of Girshick\[10\] and Zhou et al.\[33\]).
Chapter 5

Experiments and Results

In this chapter, we explain the experiments setup and dataset and discuss the results in
details. Section 5.1 overviews the dataset for the experiments. Section 5.3 provides com-
parisons between different models and in the last, ablation studies are presented in Section
5.4.

5.1 Dataset Overview

We use the PASCAL VOC 2007 dataset in the experiments. This dataset contains 20
classes for object detection task in the following 4 categories:

- Animal: bird, cat, cow, dog, horse, sheep
- Vehicle: aeroplane, bicycle, boat, bus, car, motorbike, train
- Furniture: bottle, chair, dining table, potted plant, sofa, tv/monitor
- Person: person

The dataset has been split into two same-size halves as train/validation and test sets. Each
set contains 5011 images with 12608 object instances. The number of object instances for
each class is more or less the same in train/validation and test sets. Each object instance
has been annotated with a bounding-box and an object class (Figure 5.1).
5.2 Experiments Setup

For the experiments, we use AlexNet\cite{alexnet} instances as base networks which have been pre-trained on ImageNet\cite{imagenet}, MIT Places\cite{mit_places}, or both datasets. Then we fine-tune the architecture over the PASCAL VOC 2007 train/val set. Horizontally flipped images are also added to the dataset for data augmentation. We also subtract the pixel mean-value from the images.

Selective search algorithm\cite{selective_search} extracts 2k ROIs from each image. The minibatch size, i.e., number of ROIs per image, is 256 in our experiments. Foreground background ratio in each minibatch is 1 to 3, i.e., 25% of ROIs are foregounds and 0.5 IoU threshold is used for determining the foreground/background ROIs. All the networks are optimized with a fixed weight decay of 5e-4 and a step-wise learning strategy, i.e., 30k iterations with 1e-3 learning rate, then 10k iterations with 1e-4, and finally another 10k iteration with 1e-5 learning rate. In all experiments, the entire network is fine-tuned except the first convolutional layer unless otherwise stated.

The baseline in our studies is Fast-RCNN network\cite{fast_rcnn} which uses AlexNet pretrained on ImageNet dataset. As mentioned in Chapter 4, we proposed three different architectures: Implicit Context Network (ContextNet-I), Early-Fusion Explicit Context Network (ContextNet-X EF), and Late-Fusion Explicit Context Network (ContextNet-X+ LF). All these architectures exploit pretrained AlexNet as underlying network. Both ContextNet-X EF and ContextNet-X+ LF benefit from AlexNet pretrained on MIT Places dataset for context cue extraction. ContextNet-I and ContextNet-X+ LF employ an AlexNet pretrained on both ImageNet and MIT Places for object feature extraction and classification.

5.3 Results

The results for the baseline and proposed networks on the PASCAL VOC 2007 test set have been reported in Table 5.1. ContextNet-X+ LF outperforms all other methods in 14 out of 20 object classes. The most improvements are in the following object classes: aeroplane, boat, person, and sofa. At first glance, the improvement for the person class seems strange but if you compare Figures A.9(a) and (d), the context cues help to reduce the confusion of the person class with the background and improve the localization.
ContextNet-I has best detection results for 5 object classes. Overall ContextNet-X+ LF is the best method with 57.5 mean-average precision (mAP) and 2% improvement in mAP over the baseline.

In Figure 5.2, the detection types statistics for each object class category (Section 5.1) has been reported. The person class has been added to the animal category and reduced the number of categories to three. 4 types of false detections are shown in the figure (white represents the correct detections):

1. Background (purple): confusing background with the object class of interest
2. Similar (red): confusing similar classes with the object class of interest
3. Other (green): confusing other classes with the object class of interest
4. Localization (blue): detection error due to localization, i.e., the IoU of the predicted bounding-box is less than 0.5 threshold

ContextNet-X+ LF improves the detection accuracy for the furniture and vehicles categories by 3 percents. For the furniture category, the confusion of object with similar and other classes has been reduced. In vehicle category, the improvement comes from the reduction of error in localization and confusion with background and similar classes. For animal category, the accuracy remains the same because while we see improvements in localization but confusion with the background increases.

Figures 5.3, 5.4, and 5.5 demonstrate the trend in detection types for furniture, vehicle, and animal categories, respectively. The X-axis is total detections in decreasing detection score order and Y-axis shows the percentage of each detection types. The red solid (dashed) line represents the recall while the number of detection increases if a strong (weak) localization imposed, i.e., the IoU threshold set to 0.5 (0.1). In the furniture category, ContextNet-X+ LF improves the correct detections for not-easy objects while confuses the very easy objects more. Also, you can see in Figure 5.3(d), the recall with strong localization measure increases. For the vehicle category, the localization error slightly decreases meanwhile the recall increases in both weak and strong localization metrics. ContextNet-X+ LF, in comparison with the baseline, decreases the localization error for easy and medium object instances in the animal category (Figure 5.5 (a) and (d)) but adds more confusion between similar object classes. Therefore, the overall accuracy remains the same for this category.

We observe, in general, the context cues help the network to improve detection performance. For medium and hard object instances, context decreases the localization error and reduces the confusion of object with similar classes and background. But it reduces the accuracies for very easy object instances in some cases.
Table 5.1: Results on the PASCAL VOC 2007 (precision reported). All networks have been trained on the PASCAL VOC 2007 trainval set. ContextNet-I is an implicit context network while ContextNet-X EF and ContextNet-X + LF are the explicit ones. ContextNet-X EF/ContextNet-X + LF utilizes an/a early/late-fusion architecture. ContextNet-X + uses an AlexNet which has been trained on both ImageNet and MIT places datasets as the object feature extractor.


Table 5.3: Explicit context cues effect on models’ precision on the PASCAL VOC 2007[7] test set. ContextNet-I and ContextNet-X + are an implicit and a late-fusion explicit context network, respectively. Both networks use an AlexNet for object feature extraction which has been trained on both ImageNet and MIT Places datasets. ContextNet-X + benefits from a context extraction module. Both networks have been trained on the PASCAL VOC trainval set.
Figure 5.2: Detection type statistics for different sets of classes using [12]. Each plot shows the percentage of each detection type for each network and object category set. Detection types are as follows: Correct detection (white), Localization -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Similar -false positive due to confusion with similar object classes- (red), Other -false positive due to confusion with dissimilar object classes- (green), Background -false positive due to confusion with background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively. The late-fusion architecture increases the detection accuracies in the most cases.
Figure 5.3: Distribution of detection type for Furniture classes using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
Figure 5.4: Distribution of detection type for Vehicle classes using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
Figure 5.5: Distribution of detection type for Animal classes using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
5.4 Ablation studies

We perform ablation studies in this section to better understand the effect of different design choices in the proposed networks. The following options are examined: (1) explicit and implicit context networks (Subsection 5.4.1); (2) explicit context signal (5.4.2); (3) fine-tuning (5.4.3); (4) using scene classification as context cues vs. scene embedding (5.4.4); (5) using scene classification network as context extraction module vs. scene-object classification network (5.4.5).

5.4.1 Explicit vs. implicit context networks

We can model the context implicitly (ContextNet-I) or explicitly using early-fusion approach (ContextNet-X EF). The comparison is made in Table 5.2. For all object classes but two, ContextNet-I outperforms ContextNet-X EF and its mAP is 2.7% higher. Because we concatenate scene and object features after ROI pooling in the early-fusion networks, the input size of the next FC layer is changed and we need to train it from scratch. The total number of weights in this layer is 38.5 million. Therefore, the FC layer over-fits to our small data and cannot generalize well to the test set.

5.4.2 Effect of explicit context signal

We observed, in the previous experiment, the implicit context network achieves a better mAP than the explicit one. In addition to over-fitting, we know that when we have more data and classes to distinguish, the resulting CNN learns better discriminative features [3]. Now, we add the explicit context signal to ContextNet-I using the late-fusion mechanism (ContextNet-X+ in Table 5.3). As you can see in Table 5.3 adding an explicit signal helps the network in most cases (12 out of 20 classes).

5.4.3 Effect of fine-tuning

Till now, we did not fine-tune the context extraction module over the PASCAL VOC dataset. Now, we fine-tune this module in ContextNet-X+ and reach “ContextNet-X+ FT” in Table 5.4. The results indicate that fine-tuning the context module helps in almost all classes and improves the mAP by around 1%.
5.4.4 Scene classification vs. scene embedding

We already added the scene classification probabilities (205-dimensional vector) as context signal in the previous experiments. Another way of integrating context cues is to use the layer before the last one as context (4069-dimensional vector) in ContextNet-X+. This 4069-dimensional vector is a scene embedding while the previous one (205-dimensional vector) was a scene classification. The resulting network is shown as “ContextNet-X+ Em” in Table 5.5. As you can see, the scene classification have a better performance than the scene embedding approach. We can justify the results in three ways. First, the scene classification provides explicit cues about the type of environment so provides better priors for object detection. Second, we get better features because we use deeper features in the network[3]. Third, the dimension of context features is much lower in comparison with embedding (1/20), therefore the number of learned parameters in classifier and regressor is much lesser than the embedding approach (about half a million).

5.4.5 Scene classification vs. scene-object classification

In the last experiments, we study the effect of type of context cues in the object detection performance. For doing so, we replace the context extraction module of ContextNet-X+ with another AlexNet which is pretrained on both ImageNet and MIT Places datasets. This network can provide cues about the presence of objects in addition to scene environment. The proposed network is called “ContextNet-X+ Hyb” in Table 5.6. The observed results demonstrate that the environment type is a better context cue than the environment-object one.
<table>
<thead>
<tr>
<th>methods</th>
<th>aero</th>
<th>bike</th>
<th>bird</th>
<th>boat</th>
<th>bottl</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
<th>tabl</th>
<th>dog</th>
<th>hors</th>
<th>mbik</th>
<th>prsn</th>
<th>plnt</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ContextNet-X+</td>
<td>65.2</td>
<td>68.5</td>
<td>48.2</td>
<td>38.8</td>
<td>22.4</td>
<td>67.5</td>
<td>69.1</td>
<td>66.9</td>
<td>30.1</td>
<td>59.8</td>
<td>63.2</td>
<td>62.0</td>
<td>72.7</td>
<td>68.1</td>
<td>57.4</td>
<td>26.8</td>
<td>51.2</td>
<td>58.0</td>
<td>74.6</td>
<td>60.4</td>
<td>56.6</td>
</tr>
<tr>
<td>ContextNet-X+ FT</td>
<td>66.4</td>
<td>68.1</td>
<td>51.2</td>
<td>43.4</td>
<td>23.0</td>
<td>67.9</td>
<td>70.4</td>
<td>68.4</td>
<td>30.4</td>
<td>60.2</td>
<td>59.9</td>
<td>63.5</td>
<td>72.4</td>
<td>68.9</td>
<td>59.1</td>
<td>27.6</td>
<td>53.1</td>
<td>61.5</td>
<td>72.1</td>
<td>62.0</td>
<td>57.5</td>
</tr>
</tbody>
</table>

Table 5.4: Fine-tuning effect on models’ precision on the PASCAL VOC 2007[7] test set. ContextNet-X+/ContextNet-X+ FT is a late-fusion explicit context network. Both networks use an AlexNet for object feature extraction which has been trained on both ImageNet and MIT Places datasets. In ContextNet-X+, the context extraction part has been frozen but in ContextNet-X+ FT is fine-tuned on the PASCAL VOC 2007 trainval set.

<table>
<thead>
<tr>
<th>methods</th>
<th>aero</th>
<th>bike</th>
<th>bird</th>
<th>boat</th>
<th>bottl</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
<th>tabl</th>
<th>dog</th>
<th>hors</th>
<th>mbik</th>
<th>prsn</th>
<th>plnt</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ContextNet-X+ Em</td>
<td>65.7</td>
<td>70.6</td>
<td>49.4</td>
<td>42.6</td>
<td>22.9</td>
<td>63.6</td>
<td>70.0</td>
<td>68.0</td>
<td>31.6</td>
<td>61.5</td>
<td>64.0</td>
<td>63.9</td>
<td>68.6</td>
<td>58.6</td>
<td>27.7</td>
<td>48.3</td>
<td>59.2</td>
<td>70.7</td>
<td>60.2</td>
<td>56.8</td>
<td></td>
</tr>
<tr>
<td>ContextNet-X+</td>
<td>66.4</td>
<td>68.1</td>
<td>51.2</td>
<td>43.4</td>
<td>23.0</td>
<td>67.9</td>
<td>70.4</td>
<td>68.4</td>
<td>30.4</td>
<td>60.2</td>
<td>59.9</td>
<td>63.5</td>
<td>72.4</td>
<td>68.9</td>
<td>59.1</td>
<td>27.6</td>
<td>53.1</td>
<td>61.5</td>
<td>72.1</td>
<td>62.0</td>
<td>57.5</td>
</tr>
</tbody>
</table>

Table 5.5: Scene classification & scene embedding precision comparison on the PASCAL VOC 2007[7] test set. ContextNet-X+ Em/ContextNet-X+ is an explicit context network. Both networks have been trained on the PASCAL VOC trainval set and use the late-fusion architecture. ContextNet-X+ Em uses the FC7 layer of AlexNet as context cues instead of FC8.

<table>
<thead>
<tr>
<th>methods</th>
<th>aero</th>
<th>bike</th>
<th>bird</th>
<th>boat</th>
<th>bottl</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
<th>tabl</th>
<th>dog</th>
<th>hors</th>
<th>mbik</th>
<th>prsn</th>
<th>plnt</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ContextNet-X+ Hyb</td>
<td>64.3</td>
<td>69.3</td>
<td>45.7</td>
<td>42.2</td>
<td>21.9</td>
<td>68.8</td>
<td>69.0</td>
<td>67.1</td>
<td>29.1</td>
<td>62.2</td>
<td>59.4</td>
<td>60.0</td>
<td>71.7</td>
<td>67.3</td>
<td>58.2</td>
<td>25.6</td>
<td>51.0</td>
<td>56.8</td>
<td>73.3</td>
<td>58.3</td>
<td>56.1</td>
</tr>
<tr>
<td>ContextNet-X+</td>
<td>66.4</td>
<td>68.1</td>
<td>51.2</td>
<td>43.4</td>
<td>23.0</td>
<td>67.9</td>
<td>70.4</td>
<td>68.4</td>
<td>30.4</td>
<td>60.2</td>
<td>59.9</td>
<td>63.5</td>
<td>72.4</td>
<td>68.9</td>
<td>59.1</td>
<td>27.6</td>
<td>53.1</td>
<td>61.5</td>
<td>72.1</td>
<td>62.0</td>
<td>57.5</td>
</tr>
</tbody>
</table>

Table 5.6: Scene & Scene-Object classification precision comparison on the PASCAL VOC 2007[7] test set. ContextNet-X+ Hyb/ContextNet-X+ is an explicit context network. Both networks have been trained on the PASCAL VOC trainval set and use the late-fusion architecture. ContextNet-X+ Hyb uses an AlexNet which has been trained on both ImageNet and MIT Places dataset as the context extraction module.
Chapter 6

Future Work

Object detection models have been studied for a long time. Recently, by introducing huge datasets such as ImageNet, MS-COCO, MIT Places, and YouTube-8M, the object detection field has been transformed from classical handcrafted features and latent variable models to deep learning methods. There are several paths to extend and improve the proposed methods:

- evaluating the performance of proposed methods on the most recent datasets such as the MS-COCO,
- incorporating context cues to the most recently proposed approaches (e.g., SSD (Single Shot MultiBox Detector) and YOLO (Unified, Real-Time Object Detection)),
- studying the performance effect of pre-training the early-fusion explicit context network on the ImageNet and the MIT Places datasets,
- using presence of other objects and their locations as context cues for detection and post-processing the detection outputs in addition to non-maximum suppression,
- training the pipeline simultaneously to perform three tasks of object detection, scene recognition, and semantic segmentation using the MS-COCO, the MIT Places, and the ImageNet datasets.
Chapter 7

Conclusion

In this thesis, we presented three methods to utilize the context cues to improve the object detection pipelines: The explicit context networks which explicitly model the context in the object detection frameworks by using early and late-fusion approaches. The experiments and results have been reported for the PASCAL VOC 2007 dataset. We comprehensively studied the different design choices for training procedure such as fine-tuning as well as different types of context cues (e.g., scene classification probabilities, scene embedding, and object-scene probabilities). The final proposed method enhanced the detection result by 2 percent over the baseline methods and outperforms in 14 object classes out of 20. By performing a thorough analysis of detections, we found out the improvement caused by reducing the confusion of object of interest with similar and background classes and also improving the localization.
Bibliography


Appendix A

More Results’ Analysis

A.1 Detection Types

Figure A.1: Detection type statistics for Aeroplane class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
Figure A.2: Detection type statistics for Boat class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

Figure A.3: Detection type statistics for Car class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
Figure A.4: Detection type statistics for Person class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X* LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.

Figure A.5: Detection type statistics for TV/Monitor class using [12]. Each plot shows the percentage of each detection type for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X* LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
A.2 Detection Trends
Figure A.6: Distribution of detection type for Aeroplane class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
Figure A.7: Distribution of detection type for Boat class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
Figure A.8: Distribution of detection type for Car class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
Figure A.9: Distribution of detection type for Person class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.
Figure A.10: Distribution of detection type for TV/Monitor class using [12]. Each plot shows the percentage of each detection type as the prediction score decreases for each network. Detection types are as follows: Correct detection (white), Loc -false positive due to localization error, i.e., the IoU is less than 0.5 - (blue), Sim -false positive due to confusion with similar object classes- (red), Oth -false positive due to confusion with dissimilar object classes- (green), BG -false positive due to background- (purple). The red solid (dashed red) line shows recall as the number of objects increases based on a strong (weak) localization constraint, i.e., 0.5 (0.1) IoU threshold. Fast-RCNN uses AlexNet pretrained on ImageNet. ContextNet-I, ContextNet-X EF, and ContextNet-X+ LF are an implicit, early-fusion explicit, and late-fusion explicit context network, respectively.