Dense Image Labeling using Deep Learning

by

Md Amirul Islam

A thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada

Thesis advisor

Dr. Yang Wang & Dr. Neil Bruce

Author

Md Amirul Islam

Dense Image Labeling using Deep Learning

Abstract

Recently there has been remarkable success in pushing the state of the art in dense image labeling tasks. Most of the improvements are driven by employing end-to-end deeper feed-forward networks. First, we propose a dense image labeling approach based on Deep Convolutional Neural Networks coupled with a support vector classifier. However, in many cases precisely detecting smaller and thinner object details require representation of fine details. To overcome this limitation, we propose end-to-end encoder-decoder networks that initially make a coarse-grained prediction which is progressively refined to recover spatial details. This is achieved by gate units proposed in this thesis, that control information passed forward in order to resolve ambiguity. Furthermore, we propose an end-to-end salient object detection network that employs recurrent refinement to generate a saliency map in a coarse-to-fine fashion. Experimental results demonstrate the superiority and effectiveness of our proposed approaches.

Contents

Abstract ii
Table of Contents iv
List of Figures v
List of Tables ix
Acknowledgments xi
Dedication xii
Publications xiii

1 Introduction **1**
1.1 Contributions 4
1.2 Thesis Organization 5

2 Related Work **6**
2.1 Semantic Segmentation 6
2.2 Salient Object Detection 9

3 Dense Image Labeling Using Deep Convolutional Neural Networks **12**
3.1 Deep Convolutional Neural Network 14
3.2 SVM Learning 15
3.3 Implementation 18
3.3.1 Dataset 19
3.4 Experimental Evaluation 19
3.4.1 Evaluation on Stanford background dataset 20
3.4.2 Evaluation result on PASCAL VOC 2012 23

4 Gated Feedback Refinement for Coarse-to-Fine Dense Image Labeling **25**
4.1 Background 29
4.2 Label Refinement Network 31
4.3 Gated Feedback Refinement Network 34
4.3.1 Gate Unit 36

4.3.2 Gated Refinement Unit 38
4.3.3 Stage-wise Supervision 39
4.4 Experiments 40
4.4.1 Implementation Details 40
4.4.2 CamVid 42
4.4.3 PASCAL VOC 2012 43
4.4.4 Horse-Cow Parsing Dataset 47
4.4.5 PASCAL-Person-Part 48
4.4.6 SUN RGB-D 49
4.4.7 Ablation Analysis 51
4.4.8 Exploration Studies 52

5 Salient Object Detection **57**
5.1 Context-Aware Refinement Network 58
5.1.1 Overview 58
5.1.2 Context-Aware Refinement Unit 59
5.1.3 Saliency Prior Map 62
5.1.4 Training with Multi-stage Supervision 63
5.2 Experiments and Results 63
5.2.1 Implementation Details 64
5.2.2 Datasets and Evaluation Metrics 65
5.2.3 Performance Comparison with State-of-the-art Methods 66
5.2.4 Comparison with Different Variants 68

6 Conclusion and Future Work **70**

List of Figures

1.1 First row: images second row: predictions of our approaches. Our goal is to accurately label each pixel in the image. 3

3.1 An overview of our proposed approach. Leftmost images are samples from different datasets. From each dataset, we learn a deep convolutional neural network (DCNN). The architecture of the DCNN is shown in the middle of the figure and is described in detail in Sec. 3.1. Convolution, pooling and soft-max layers in the DCNNs are shown in different colors. ReLu layers are omitted from the box. The DCNN learned from each dataset will produce a dense labeling for a given image. We concatenate the outputs from these DCNNs to form a feature vector for each pixel in the image. We then train an SVM classifier based on these feature vectors to obtain the final labeling of each pixel in the image. 13

3.2 (a) Test image. Predicted labels produced by the classifier are used to train image specific classifier again along with the corresponding test image. Note that the known label values themselves are not used in training, but rather the labels produced by the *generic* SVM are assumed to be mostly correct and define the image specific ground truth for subsequent SVM training based on 1 image. (b) Output image. 17

3.3 Sample results of semantic segmentation on the Stanford background dataset [1]. 1st row: test images; 2nd row: ground-truth semantic segmentations; 3rd-row: segmentation results produced by ConvNet-CSVM2. Different semantic classes are represented by different colors. 22

3.4 Sample results of geometric labeling on the Stanford background dataset [1]. 1st row: test images; 2nd row: ground-truth geometric labels; 3rd-row: geometric labeling results produced by ConvNet-CSVM2. Different geometric classes are represented by different colors. 22

3.5 Sample results of semantic segmentation on the PASCAL VOC 2012 dataset [2]. 1st column: test images; 2nd column: ground-truth semantic segmentations; 3rd-column: segmentation results produced by ConvNet-CSVM2. Different semantic classes are represented by different colors. 24

4.1 Labeling objects such as the column-pole, pedestrian or bicyclist shown above requires low-level finer details as well as high-level contextual information, despite of varying light scenes. In this work, we propose a gated feedback refinement network, which can be used with any bottom-up, feed-forward ConvNet. We show that the finer features learnt by our approach lead to significantly improved semantic segmentation. 26

4.2 An illustration of the relationship between receptive field size across layers, and ambiguity that may arise. In this case, the larger (and more discriminative) receptive field (blue) resides at a deeper layer of the network, and may be of value in refining the representation carried by an earlier layer (orange) to resolve ambiguity and improve upon labeling performance. 27

4.3 Overview of our Label Refinement network and Gated Feedback Refinement Network (G-FRNet). In LRN, each of the decoder layers has long range connections to its corresponding encoder layer whereas G-FRNet adds a gate between the long range connections to modulate the influence of these connections. 31

4.4 Overview of our Gated Feedback Refinement Network (G-FRNet). We use feature maps with different spatial dimensions produced by the encoder (f_1, f_2,..., f_7) to reconstruct a small (i.e. coarse) label map Pm^G. The decoder progressively refines the label map by adding details from feature maps in the encoder network. At each stage of decoding, a refinement unit (RU_1, RU_2,..., RU_5) produces a new label map with larger spatial dimensions by taking information from the previous label map and encoder layers as inputs (denoted by the edge connecting G_i and RU_i). The main novelty of the model is that information from earlier encoder layers passes through a gate unit before being forwarded to the decoder. We use standard 2x bilinear upsampling on each class score map before passing it to the next stage refinement module. We also use down-sampled ground-truth label maps to provide supervision (l_1, l_2, ..., l_6) at each decoding stage. 32

4.5 Detailed overview of a Gated Refinement Unit. The refinement unit is unfolded here for i^{th} stage. The refinement module (similar to [3]) is composed of convolution, batch normalization, concatenation, and upsampling operations. 37

4.6 Visualization of hierarchical gated refinement scheme. The refinement process integrates higher-frequency details with the lower resolution label map at each stage. Class-wise activation maps for each gate are shown as heatmaps. 40
4.7 Qualitative results on the CamVid dataset. G-FRNet is capable of retaining the shape of smaller and finer object categories (e.g. column-pole, side-walk, bicyclist, and sign-symbols) accurately compared to FSO [4]. 44
4.8 Qualitative comparisons between the FCN[5] model and our LRN model on the PASCAL VOC 2012 dataset. 45
4.9 Qualitative results on Horse-Cow parsing dataset [6]. 48
4.10 Stage-wise visualization of semantic segmentation results on PASCAL VOC 2012. For each row, we show the input image, ground-truth, and the prediction map produced at each stage of our feedback refinement network. 50
4.11 Comparison of stage-wise mean IoU on (a) CamVid dataset; (b) PASCAL VOC 2012 validation set (c) Horse parsing (d) Cow parsing (e) Pascal-Person-Part and (f) SUN-RGBD dataset between LRN [3] and proposed network G-FRNet. Note that Pascal-Person-Part stage-wise results are using G-FRNet-Res101 and LRN-Res101. 52
4.12 Class-wise heatmap visualization on PASCAL VOC 2012 validation set images after each stage of refinement. Interestingly, the network gradually aligns itself more precisely with semantic labels, while correcting initially mislabeled regions. The rightmost column shows the heatmap of the final prediction layer. 53
4.13 Analysis on the number of model parameters (in millions) and the mean IoU (%) on PASCAL VOC 2012 validation set for different methods. The rightmost method is our proposed model, which achieves best performance, even with considerably fewer parameters, and a more parsimonious model structure. 54
4.14 Qualitative analysis of dense predictions on a few challenging object categories in CamVid dataset. For example, pedestrians are labeled accurately by our model, despite the varying light settings. 55

5.1 An example of applying context-aware refinement network to an initial saliency map produced by the encoder network. Compared to initial saliency map, the refined saliency map has significantly sharper edges and also has better spatial information. 58

5.2 An overview of the proposed salient object detection framework. The bottom-up (encoder) network consisting of multiple layers (e.g. convolution, batch normalization, pooling, ReLU) is integrated with the refinement network through skip connections. The refinement network has context-aware refinement units (CRU_1, CRU_2,..., CRU_4) that take a bottom-up feature map and previous stage prediction map (S_i) to generate subsequent stage prediction maps (S_{i+1}). We down-sampled the ground-truth saliency maps to incorporate stage-wise supervision (l_1, l_2,..., l_6) in the refinement network. GCN and BR are used in CRUs (see main text for details). Note that we also combine the final prediction map with the saliency prior to obtain the final saliency map. 59

5.3 Visual comparison of saliency maps with state-of-the-art methods, including our approach. (a) Input image (b) ground truth (c) CARNet (d) DCL (e) DHS (f) DSR (g) DRFI (h) HS (i) HDCT (j) MC. Our approach consistently produces saliency maps closest to the ground truth. 67

5.4 Comparison with state-of-the-art salient object detection methods on 3 different datasets. For each dataset, the first row shows the precision-recall curves and second row shows the F-measure and AUC. Our proposed approach CARNet† consistently outperforms other methods across all the datasets. In particular, the PR-curves show that our approach achieves significantly higher precision with higher recall, which demonstrates that our method locates salient objects more accurately and precisely. Note that DHSNet [7] includes the test set of DUT-OMRON in its training data. Therefore, we do not include it in the comparison based on the DUT-OMRON dataset. 68

List of Tables

3.1 Details of the architecture of the convolutional neural network. 14
3.2 Description of different configurations used in our experiments. . . . 18
3.3 Quantitative results of different approaches for the semantic segmentation task on the Stanford background dataset [1]. We show the accuracy for each semantic class, the average accuracy of these eight classes (MCA), and the overall pixel accuracy (overall). 21
3.4 (a) Quantitative results of different approaches for the geometry labeling task on the Stanford background dataset [1]. (b) Comparison with Gould et al. [1] on the geometric labeling task on the Stanford background dataset. (c) Comparison with state-of-the-art semantic segmentation approaches on Stanford background (semantic) dataset [1]. 21
3.5 Quantitative results of different approaches for the semantic segmentation task on the PASCAL VOC 2012 dataset [2]. 23

4.1 Quantitative results on the CamVid dataset [64]. We report per-class IoU and mean IoU for each method. We split methods into two categories depending on whether or not they use ConvNet. Not surprisingly, ConvNet-based methods typically outperform non-ConvNet methods. Our approach achieves state-of-the-art results on this dataset. Note that the improvements on smaller and finer objects are particularly pronounced for our model. 42
4.2 Comparison of different methods on PASCAL VOC 2012 validation set. Note that DeconvNet [8] result is taken from [9]. 44
4.3 Quantitative results in terms of mean IoU on PASCAL VOC 2012 test set. Note that G-FRNet-Res101 includes CRF. 45
4.4 Comparison of object parsing performance with state-of-the-art methods on Horse-Cow parsing dataset [6]. 47
4.5 (a) Comparison of object parsing results with other state-of-the-art results on Pascal Person-Part dataset [6]. (b) Qualitative results on the Pascal Person-Part dataset. 49

4.6 (a) Comparison of scene parsing results with other state-of-the-art results on SUN RGB-D dataset [10]. (b) Qualitative results on the SUN RGB-D dataset. 50
4.7 (a) Stage-wise mean IoU on PASCAL VOC 2012 validation set, CamVid, and SUN-RGBD dataset. (b) Stage-wise mean IoU on Horse-Cow parsing and Pascal-Person-Part dataset. Note that for Pascal-Person-Part stage-wise numbers are reported for G-FRNet-Res101 and LRN-Res101 52

5.1 Quantitative comparison (in terms of average F_β and MAE) with state-of-the-the methods. Best and second best scores are shown in red and blue text respectively. 66
5.2 Comparison of different variants of our proposed approach. Our final model CARNet† achieves the best performance when compared to other variants of our model. 69

Acknowledgments

I consider myself very lucky and honored to have so many amazing people leading and pushing me through in the accomplishment of this thesis. First and foremost, I would like to express my sincere gratitude to my admirable supervisors, Dr. Yang Wang and Dr. Neil Bruce, for whom I was able to indulge myself in the research of Deep Learning. They have truly been an inspiration for me and I have learned quite a lot from them. Their thoughtful reviews and suggestions during my research and paper writings have made my expedition outstanding. I could not ask for more and would like to take this opportunity to say a humble 'Thank you' to both of you, Yang and Neil!

I would also choose this moment to thank my committee members, Dr. Carson Kai-Sang Leung and Dr. Ekram Hossain, who have provided valuable suggestions and constructive feedback in perfecting my thesis.

I would also like to recognize the funding sources - The University of Manitoba, Faculty of Science for their continuous financial support.

I would like to take this opportunity to express my gratitude to my wonderful lab mates for their time, support and suggestions.

Also, this report, and indeed the completion of this Master of Science, would not have been possible without the love, eternal patience and support of my wonderful family members.

Finally, but by no means the least, I would like to recognize the continuous motivation and support from my acquaintances in Dhaka, Winnipeg and at the University of Manitoba an incredible group of people who made my journey an enjoyable one.

This thesis is dedicated to my parents
for their love, endless support
and encouragement.

Publications

Some of the ideas, materials and figures in this thesis have appeared previously in the following publications, pre-prints, and submitted manuscripts:

1. **M. A. Islam**, N. Bruce, and Y. Wang. Dense Image Labeling using Deep Convolutional Neural Networks. *Conference on Computer and Robot Vision (CRV)*, Victoria, Canada, June 2016.

2. **M. A. Islam**, S. Naha, M. Rochan, N. Bruce, and Y. Wang. Label Refinement Network for Coarse-to-Fine Semantic Segmentation. ArXiv preprint arXiv:1703.00551, 2017.

3. **M. A. Islam**, M. Rochan, N. Bruce and Y. Wang. Gated Feedback Refinement Network for Dense Image Labeling. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Hawaii, USA, July 2017.

4. **M. A. Islam**, M. Rochan, S. Naha, N. Bruce and Y. Wang. Gated Feedback Refinement Network for Coarse-to-Fine Dense Image Labeling. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017. (Under review)

5. **M. A. Islam**, M. Kalash, M. Rochan, N. Bruce and Y. Wang. Salient Object Detection using a Context-Aware Refinement Network. *British Machine Vision Conference (BMVC)*, London, 2017.

Chapter 1

Introduction

In recent years, there have been significant advances in deep learning applied to problems in computer vision. This has been met with a great deal of success, and has given rise to proliferation of significant variety in the structure of neural networks. Many current deep learning models apply a cascade comprised of repeated convolutional stages, followed by spatial pooling. Down-sampling by pooling allows for a very large pool of distinct and rich features, albeit at the expense of spatial resolution. For recognition problems, the loss of spatial precision is not especially problematic. However, dense image labeling problems (e.g. semantic segmentation, semantic object parsing, salient object detection) require pixel-level precision.

In order to properly train deep convolutional neural networks, one typically needs a large amount of labeled training data. Compared with image classification, training data for dense image labeling tasks is much more onerous to produce. For example, the current semantic segmentation datasets are orders of magnitude smaller than datasets that address the problem of image classification. Most DCNN-based seman-

tic segmentation methods use pre-trained image classification models and fine-tune those models for semantic segmentation.

Deep learning models for dense image labeling problems typically involve a decoding process that gradually recovers a pixel level specification of categories. In some cases this *decoding* is done in one step [5; 11], while in other instances, both the encoding of patterns, and gradual recovery of spatial resolution are hierarchical. It is interesting to note that this mirrors the observed computational structure of human vision wherein space is abstracted away in favour of rich features, and recognition of patterns precedes their precise localization [12]. Some models that have shown success for segmentation problems [13; 8] share a common structure involving stage-wise encoding of an input image, followed by stage-wise decoding to recover a per-pixel categorization. At an abstract level, this is reminiscent of a single network that involves a feed-forward pass, followed by a recurrent pass from the top layer downward where additional computation and refinement ensues. There are tangible distinctions though, in that decoding is typically driven only by information flow that satisfies solving a specific labeling problem, and that all decoding may be informed only by the representation carried by the highest encoder layer.

Based on the above limitations and observations, we propose simple and effective methods to densely label semantic or salient objects in the image. Figure 1.1 illustrates the goal of our work.

We propose a DCNN based dense image labeling approach that leverages different types of representations for predicting class labels, that are not directly related to the target task. Our work is motivated by the aforementioned observation. Even though

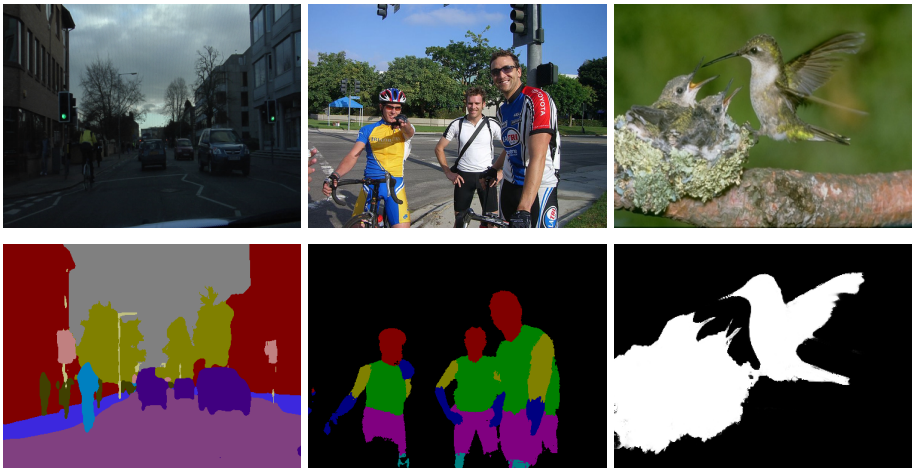


Figure 1.1: First row: images second row: predictions of our approaches. Our goal is to accurately label each pixel in the image.

the annotations on different datasets are not compatible, the visual representations that are of value in solving these different dense labeling tasks may overlap. In making use of these diverse datasets to learn a good visual feature representation, this may present the opportunity for better performance for any specific dense labeling task corresponding to any of the problems associated with a specific subset of data corresponding to this larger dataset comprised of heterogeneous and incompatible labels.

Then we develop two encoder-decoder based deep learning architectures that address the limitation of combining local and global contextual information. Finally, inspired by the success of encoder-decoder networks in semantic segmentation task, we apply a network with this structure to detect salient regions in an end-to-end fashion that employs recurrent refinement to generate a saliency map in a coarse-to-fine fashion by incorporating finer details in the detection framework.

1.1 Contributions

We summarize our main contributions as follows:

- We propose a new approach for dense image labeling by taking advantage of multiple datasets, wherein class labels corresponding to different datasets might not be compatible. Experimental results demonstrate the utility of making use of intelligence tied to different sources of labeling in improving upon baseline performance.
- We propose a new perspective on semantic segmentation, or more generally, pixel-wise labeling of images. Instead of predicting the final segmentation result in a single shot, we propose to solve the problem in a coarse-to-fine fashion by first predicting a coarse labeling, then progressively refining this coarse grained prediction towards finer scale results. We introduce a novel gating mechanism to modulate how information is passed from the encoder to the decoder in the network. The gating mechanism allows the network to filter out ambiguity concerning object categories as information is passed through the network. The proposed approach is the first that uses a gating mechanism in an encoder-decoder framework for the task of semantic segmentation in order to combine local and global contextual information. Unlike most of the previous methods that only have supervision at the end of their network, our model has supervision at multiple resolutions in the network.
- We introduce a novel end-to-end encoder-decoder based salient object detection model that can simultaneously predict saliency maps at different resolutions. We propose a context-aware refinement network, which serves as the decoder

network, and can hierarchically and progressively refine saliency maps to recover fine details of the image by integrating local and global contextual information through gate units, global convolution units and boundary refinement units. Moreover, we combine the prior map with the final prediction map.

- Experimental results on benchmark datasets and comparisons with recent state-of-the-art approaches demonstrate the effectiveness and superiority of our approaches on the several dense image labeling tasks.

1.2 Thesis Organization

The remainder of the thesis is organized as follows. In Chapter 2, we briefly discuss related work. In Chapter 3, we describe a dense image labeling approach based on deep convolutional neural networks coupled with a support vector classifier. We also show the strength of leveraging different types of representations for predicting class labels, that are not directly related to the target task (e.g. predicted scene geometry may help assigning object labels). In Chapter 4, we develop two encoder-decoder based deep learning architectures to address the dense image labeling problem. In Chapter 5, we propose a novel end-to-end encoder-decoder based salient object detection model. In Chapter 6, we conclude this thesis and discuss possible future directions.

Chapter 2

Related Work

In this chapter, we briefly discuss the works most related to our proposed methods.

CNNs have shown tremendous success in various visual recognition tasks, e.g. image classification [14], object detection [15] and action recognition [16]. Recent approaches have also shown enhanced discriminative power of features within CNNs by increasing the depth of the network [17; 18; 19]. Dense labeling tasks, such as semantic segmentation, semantic object parsing, salient object detection have also benefited from such deep networks. Recently, there has been work on adapting CNNs for pixel-wise image labeling problems such as semantic segmentation [13; 11; 20; 21; 5; 22; 8; 23], salient object detection [79; 52; 7; 53; 89; 51].

2.1 Semantic Segmentation

Semantic segmentation is fundamental to image understanding as it assigns class labels to individual pixels in an image. The problem of semantic segmentation has

been studied over decades but remains a challenging task. Some of the difficulty is due to variation in the appearance of objects, background clutter, pose variations, scale changes, occlusions, and other factors.

Fully Convolutional Neural Networks(F-CNN) based approaches carry a major limitation is that they produce a segmentation map of relatively low spatial resolution. There exist a number of methods that address this limitation by generating segmentation maps of higher spatial resolution. Long et al. [5] propose the first semantic segmentation network that trained fully convolutional networks (FCN) in an end-to-end fashion to accomplish pixel-wise prediction corresponding to a whole image. Their network is based on VGG-16 [17]. They define a novel skip architecture that combines semantic information with deep, coarse, and appearance based information. Recent methods including DeepLab-CRF [11] and DeepLabv2 [24] predict a mid-resolution label map by controlling the resolution of feature responses within the network, then directly upsampling to the original spatial resolution of the image by bi-linear interpolation. Finally a dense CRF is applied on top of the final prediction to refine boundaries. CRF-RNN [25] extends the DeepLab [11] network to include end-to-end learning of the CNN and dense CRF. Recently, Yu et al. [26] introduce dilated convolution to expand the effective receptive field of feature maps to encode extra contextual information within local features that brought significant benefit to the ConvNet architectures in terms of performance.

Directly related to our work is the idea of multi-scale processing in computer vision. Early work on Laplacian pyramids [27] is a classic example of capturing image structures at multiple scales. Eigen et al. [28] use a multi-scale CNN for

predicting depths, surface normals and semantic labels from a single image. Denton et al. [29] use a coarse-to-fine strategy in the context of image generation. Honari et al. [30] combine coarse-to-fine features in CNNs for facial keypoint localization. The Hypercolumn [21] method leverages features from intermediate layers to generate final predictions via stage-wise training. A recent pixel-wise labeling architecture named PixelNet [31] follows the hypercolumn strategy to combine contextual information. SegNet [13] and DeconvNet [8] apply skip connections in the form of deconvolution layers to exploit the features produced in the encoder stage to refine predictions at the time of decoding.

Few existing methods have used the idea of defining loss functions at different stages in the network to provide additional supervision in learning, and this is a characteristic of the models proposed in this thesis. The Inception model [18] uses auxiliary classifiers at the lower stages of the network to encourage the network to learn discriminative features. Lee et al. [32] propose a deeply-supervised network for image classification. Xie et al. [33] use a similar idea for edge detection.

Architecturally, our encoder-decoder network based works are closest to the strategies proposed in a few existing works [34; 35; 36; 37; 38] in which coarse-grained prediction maps are refined to generate final predictions by top-down modulation. However, how to effectively integrate fine grained simplistic features with coarse-level complex features still remains a open question. Pinheiro et al. [36] propose a ConvNet architecture that has refinement modules with top-down modulation and skip connections to refine segmentation proposals. Ranjan et al. [38] also propose a spatial pyramid network that initially predicts a low-resolution optical flow map and

iteratively uses the lower layer information to obtain high-resolution optical flow.

There is strong evidence of top-down modulation and feedback refinement in neural information processing within human and primate visual pathways [39; 40; 41; 42], wherein more precisely localized representations that may be ambiguous are modulated or gated by higher-level features, which also act as attentional mechanism for selecting unambiguous features. Our proposed model is based on this intuition and integrates this style of information processing scheme within ConvNets in a fully encapsulated end-to-end trainable model.

2.2 Salient Object Detection

Over the past few years, a large number of salient object detection approaches have been developed. In general, those approaches can be classified into two main categories, i.e., either contrast-based methods that use hand-crafted features or methods that apply deep learning to learn both features and the classifier.

Contrast based methods select and combine important features to detect objects that attract attention. Some of these methods use local, low-level features such as multi-scale color, intensity and orientation filters [43], mid-level visual cues [44], or the contrast of multiple feature distributions [45]. However, other methods use global features like region descriptors [46], global region contrast [47], or a combination of features (i.e. multi-scale contrast, center surround histograms, and color spatial distributions) [48].

More recently, CNNs have shown superior performance compared to traditional methods on commonly used benchmarks. CNN based models have raised the bar

on the quality of predictions possible in multiple fields of computer vision, including salient object detection. Recently, many salient object detection methods adopt CNN based models due to the ability to extract more representative and complex high-level features. Wang et al. [50] integrate both local estimation and global search using two sequential CNNs to predict saliency maps. Local saliency information (i.e. the saliency value for each pixel) is extracted by the first CNN and then forwarded along with global contrast and geometric information to the second CNN for further refinement. Zhao et al. [51] propose a multi-context CNN that obtains and integrates global and local context information to produce saliency maps. Liu and Han [7] tackle the salient object detection problem in a global to local (coarse to fine) manner. Their architecture follows the end-to-end encoder-decoder approach where the encoder learns multiple global structured saliency cues and their optimal combination to produce a coarse saliency map. Then, another hierarchical recurrent convolutional neural network refines the coarse saliency map stage-by-stage by integrating local contextual information. Li and Yu [52] propose an end-to-end deep contrast network with two streams to enhance salient object boundary detection. They combine a pixel-level fully convolutional stream that produces a saliency map with pixel-level accuracy and a segment-wise spatial pooling stream that extracts segment-wise features. The fused saliency map is finally refined with a fully connected CRF model. Wang et al. [53] propose a recurrent fully convolutional network for saliency detection. In the first time step, they use the potential salient regions in the input image as a prior knowledge of possibly salient regions in order to predict an initial saliency map. This in turn serves as the saliency prior map for the next time step.

In contrast to the above approaches, we perform a step-by-step multi-stage supervised refinement for the encoded saliency map until the saliency map spatial resolution matches the input image. This also notably includes specific mechanisms for how early feature information is routed in making a final determination of saliency.

Although significant progress has been achieved in the last two years, there is still significant room for improvement.

Chapter 3

Dense Image Labeling Using Deep Convolutional Neural Networks

Given an input image, our goal is to produce a dense labeling of the pixels in the image. In this chapter of thesis, we consider two dense labeling tasks, namely semantic segmentation [5] and geometric labeling [1]. The goal of semantic segmentation aims to label each pixel according to the labeled object category (e.g. people, car, building, etc.). The goal of geometric labeling is to label each pixel according to its geometric class. Over the years, the computer vision community has created several benchmark datasets for these problems, but these datasets often consider different sets of classes. For example, the Stanford background dataset [1] contains 8 categorical classes (sky, tree, road, grass, water, building, mountain, foreground) and 3 geometric classes (sky, horizontal, vertical) while the PASCAL VOC data [2] contains 21 classes (20 object classes + background). In this chapter, we propose an approach for dense image labeling based on deep convolutional neural networks (DCNNs). The novelty of our

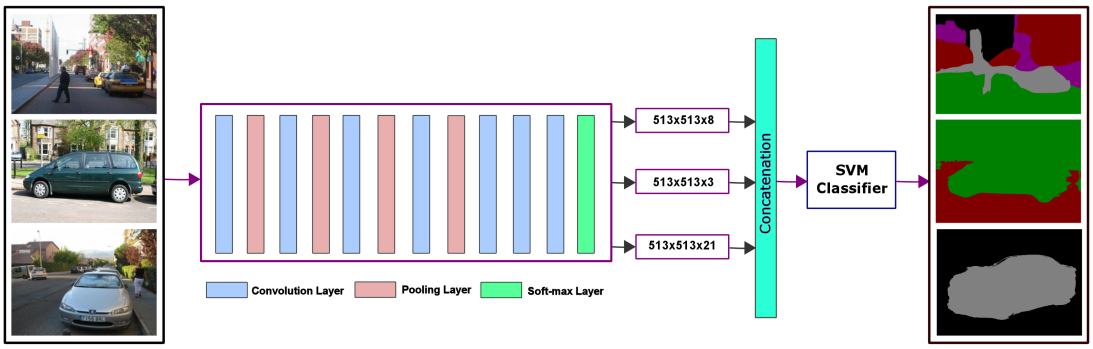


Figure 3.1: An overview of our proposed approach. Leftmost images are samples from different datasets. From each dataset, we learn a deep convolutional neural network (DCNN). The architecture of the DCNN is shown in the middle of the figure and is described in detail in Sec. 3.1. Convolution, pooling and soft-max layers in the DCNNs are shown in different colors. ReLu layers are omitted from the box. The DCNN learned from each dataset will produce a dense labeling for a given image. We concatenate the outputs from these DCNNs to form a feature vector for each pixel in the image. We then train an SVM classifier based on these feature vectors to obtain the final labeling of each pixel in the image.

approach is that we take advantage of multiple datasets even though they are defined by different sets of class labels. In particular, we use the Standard background dataset (with both semantic labels and geometric labels) and the PASCAL VOC dataset (with object class labels) in this work. An overview of our approach is illustrated in Fig. 3.1. First, we train three separate DCNNs. The first DCNN is trained on the Stanford background dataset to produce one of the 8 semantic classes for each pixel. The second DCNN is trained on the Stanford background dataset to produce one of the three geometric classes. The third DCNN is trained on the PASCAL dataset to produce one of the 20 object classes for each pixel (Sec. 3.1). For a given image, we apply these three DCNNs and concatenate their outputs to form a feature vector. We then learn a SVM classifier based on this feature vector to predict the label of each

	1	2	3	4	5	6	7	8	9	10	11	12	13
layer	2× **conv**	**max**	2 × **conv**	**max**	2× **conv**	**max**	3× **conv**	**max**	3× **conv**	**max**	**fc6**	**fc7**	**fc8**
filter-stride	3-12	3-2	3-12	3-2	3-12	3-2	3-12	3-1	3-12	3-1	3-12	1-12	1-12
#channel	64	64	128	128	256	256	512	512	512	512	1024	1024	**#label**
activation	relu	idn	relu	idn	relu	idn	relu	idn	relu	idn	relu	relu	soft
size	321	161	161	81	81	41	41	41	41	41	41	41	41

Table 3.1: Details of the architecture of the convolutional neural network.

pixel in the image (Sec. 3.2).

3.1 Deep Convolutional Neural Network

The architecture of our deep network is based on DeepLab [11], which in turn is based on the VGG-16 network [17] trained on the ImageNet classification task. In total, the network has 15 convolutional layers and 5 max-pooling layers. Table 3.1 summarizes the different layers in the network and their parameters.

An input image is passed through a stack of convolutional layers with very small kernel sizes. Spatial pooling is carried out by five max-pooling layers, which follow the convolution layers. Two fully-connected layers of VGG-16 [17] network are transformed to convolutional layers in order to get pixel-wise prediction. The last 1x1 convolution (fc8) layer is used to make sure that the number of output matches the number of labels. For example, if we train this network on the Standard background dataset to predict geometric classes for each pixel, the number of labels will be 3. If we train this network to predict object classes for each pixel, the number of labels will be 21. We use Caffe [54] for training the network.

Suppose that we want to train the network to predict semantic classes on the Stanford background dataset. There are 8 semantic categories on this dataset. Each image

is rescaled to 513x513 during training. Through convolution and pooling, the deep network extracts multi-class visual deep features and generates 8 coarse score maps. Each feature map indicates the probabilistic label map of each semantic category. The resulting feature maps are up-sampled to 513x513 using bilinear interpolation to equate the size of the input image. Therefore, the network predicts 513x513x8 labels in the end. Similarly, we will get 513x513x3 labels by training a DCNN for the geometric labeling task on the Stanford background dataset, and 513x513x21 labels by training a DCNN for the semantic labeling task on the PASCAL VOC dataset. We concatenate these three sets of features maps together in the end to get a feature map of 513x513x(8+3+21). Each pixel corresponds to a (8+3+21) dimensional feature vector in the feature map.

3.2 SVM Learning

In this section, we describe how the feature maps obtained from the DCNNs in Sec. 3.1 are processed and used to train a SVM classifier for producing the final dense labeling on a particular dataset.

For ease of presentation, let us consider the semantic segmentation problem on the PASCAL VOC dataset. This problem requires labeling each pixel as one of the 21 semantic classes defined in the PASCAL VOC. We first run the three DCNNs from Sec. 3.1 on both training and test images in the PASCAL VOC datasets (these DCNNs are trained from both the PASCAL VOC and the Standard background datasets with heterogeneous labels). Each pixel in an image is then represented by a (8+3+21=32) dimensional feature vector. We then learn a linear SVM classifier to predict the

21 semantic classes on the PASCAL VOC dataset using this 32 dimensional feature vector.

We have experimented with several approaches for constructing the training data for learning the SVM classifier.

ConvNet-SVM: This approach randomly selects a set of pixels from all the training images on PASCAL VOC. Each pixel will be a training instance with 21 dimensional feature vector from DCNN. Since we know the ground-truth semantic labels of these pixels, we can learn a SVM classifier using the ground-truth labels.

ConvNet-CSVM: This approach randomly selects a set of pixels from all the training images on PASCAL VOC. Each pixel will be a training instance with 32 dimensional feature vector, corresponding to the concatenated feature set.

Both ConvNet-SVM and ConvNet-CSVM learn the SVM classifier from the pixels sampled from the PASCAL training images. We have also experimented with sampling the pixels from the PASCAL test images.

ConvNet-CSVM2: This approach randomly selects a set of pixels from all the test images on PASCAL VOC. Each pixel will be a training instance with 32 dimensional feature vector. An image-specific SVM technique is then applied to each test image separately assuming the predicted labels (i.e. based on ConvNet-CSVM2) represent the ground-truth for that specific image. That is, given a general classifier, and associated label predictions, these may be refined by training an SVM specific to the image under consideration by making the assumption that the assigned class labels are mostly correct and treating this as the ground truth. This stage of image-specific SVM classification takes full advantage of the support vector classification stage to

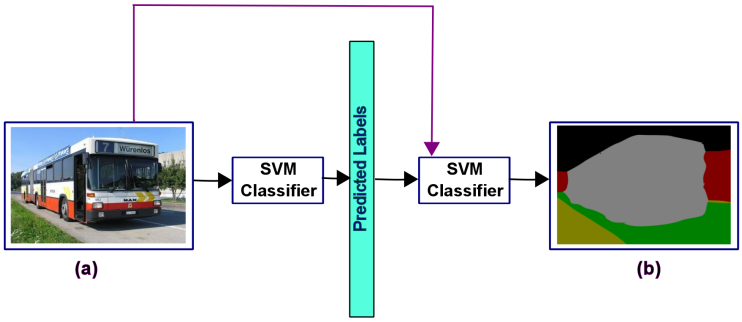


Figure 3.2: (a) Test image. Predicted labels produced by the classifier are used to train image specific classifier again along with the corresponding test image. Note that the known label values themselves are not used in training, but rather the labels produced by the *generic* SVM are assumed to be mostly correct and define the image specific ground truth for subsequent SVM training based on 1 image. (b) Output image.

refine the initial generic predictions to produce those that may better characterize a specific test image. An overview of the process is shown in Fig. 3.2

ConvNet-CSVM3: This approach is similar to Convnet-CSVM2 apart from how pixels are sampled from the test images on PASCAL VOC. We select class-wise positive pixels (those for which ConvNet-SVM produces a higher score among all other categories) with a 32 dimensional feature vector.

ConvNet-WSVM: This approach randomly selects class-wise positive pixels with a 32 dimensional feature vector from all the test images on PASCAL VOC. Then instead of linear SVM, we train a weighted SVM by assigning a different weight to each pixel to vary the contribution of each pixel in the second SVM training stage. We use the Kernel-based Possible C-means (KPCM) algorithm [55] to generate a weight for each pixel. Finally the refined pixel-wise segmentation is predicted by the refined SVM.

Table 3.2 summarizes these different approaches.

Method	Procedure
ConvNet-SVM	Trained linear SVM by choosing random samples from training images
ConvNet-CSVM	Trained linear SVM by choosing samples from training images with concatenated feature maps
ConvNet-CSVM2	Trained linear SVM by choosing samples from test images with concatenated feature maps (note that for this case and those that follow, the ground truth labels are assumed (from the preceding classifiers), and do not come from the known test image labels)
ConvNet-CSVM3	Trained linear SVM by choosing class-wise positive samples (where the DCNN produces a higher score among all other categories) from test images with concatenated feature maps
ConvNet-WSVM	Trained weighted SVM by choosing class-wise positive samples from test images with concatenated feature maps.

Table 3.2: Description of different configurations used in our experiments.

3.3 Implementation

ConvNet-CSVM model is trained and tested with Caffe [54] on a machine with 10 cores (2.3GHZ Intel Xeon E5-2630V3 CPU), 64GB RAM, 4TB hard drive, and two NVIDIA Titan X GPUs. We use some of the parameters from DeepLab [11] to initialize the network. Following [11], the batch size and initial learning rate are initialized to 30 and 0.001 respectively. We fix the momentum to 0.9, weight decay of 0.0005 and maximum iteration to 6000. The total number of parameters in the model is approximately 20.5M and training requires approximately 6 hours.

Detailed description of different configurations used in our experiments are presented in Table 3.2. Each configuration differs in the approach of creating training proposals for linear and weighted SVM. For the image specific SVM, each test image is trained separately assuming the predicted labels produced by linear or weighted SVM as ground-truth.

3.3.1 Dataset

We evaluate the proposed method on the Stanford Background Dataset [1] and the PASCAL VOC 2012 [2] segmentation challenge dataset. The Stanford background dataset contains total 715 images of urban and rural scenes. Each pixel is labeled with one of the 8 semantic classes (sky, tree, road, grass, water, building, mountain, and foreground) and one of the 3 geometric classes (sky, horizontal, and vertical). Each image is approximately 240×320 and contains at least one foreground object. The PASCAL VOC 2012 dataset [2] consists of 1464 training and 1456 test images. Each pixel in this dataset is labeled with one of the 21 categories (20 object categories and the background class).

3.4 Experimental Evaluation

We present experimental results on two different datasets: the Stanford background dataset (SBD) [1] and the PASCAL VOC 2012 [2] segmentation benchmark dataset. On the Stanford backgound dataset, we report several metrics for measuring the pixel accuracy. Let n_{ij} be the number of pixels of class i predicted to be class j, and $t_i = \sum_j n_{ij}$ be the total number of pixels of class i. Let K be the total number

of classes. We compute:

- per-class accuracy for the i-th class: n_{ii}/t_i
- average per-class accuracy: $(1/K)\sum_i n_{ii}/t_i$
- overall accuracy: $\sum_i n_{ii}/\sum_i t_i$

On the PASCAL VOC 2012 dataset, we report results using intersection-over-union (IoU) for each class and the mean IoU overall all classes as follows:

- IoU for the i-th class: $n_{ii}/(t_i + \sum_j n_{ji} - n_{ii})$
- mean IoU: $(1/K)\sum_i n_{ii}/(t_i + \sum_j n_{ji} - n_{ii})$

3.4.1 Evaluation on Stanford background dataset

In this section, we report our evaluation results on the Stanford background dataset. Following [1; 56; 22], we use 5-fold cross-validation which splits the dataset into 572 training images and 143 test images. A challenging class within this dataset is the foreground class, since it includes a wide range of objects like person, cow, bicycle, sheep, car as a singular class. The appearance of the foreground class can vary drastically across different object types. Another challenging class is the mountain class, since it appears in very few images. In order to explore the strength of leveraging different types of representations for predicting labels, we report results for different configurations. The quantitative results of semantic and geometric classes are shown in Table 3.3 and Table 3.4 (a), respectively. We can see that our method performs quite well on this dataset. Some qualitative semantic segmentation examples

Method	sky	tree	road	grass	water	building	mountain	foreground	average (%)	overall (%)
ConvNet	95.5	85.4	93.7	94.4	92.1	88.6	86.2	77.4	89.1	90.4
ConvNet-SVM	93.2	90.5	93.1	92.7	91.3	89.4	85.8	78.6	89.3	90.8
ConvNet-CSVM	94.4	85.9	95.1	91.0	92.7	90.9	85.5	86.5	89.6	91.2
ConvNet-CSVM2	93.2	90.3	96.5	92.7	92.6	96.2	65.3	75.3	87.8	**91.6**
ConvNet-CSVM3	93.9	90.1	94.4	94.2	91.7	94.2	90.7	75.4	**90.4**	90.9
ConvNet-WSVM	94.0	90.1	94.2	94.2	91.4	94.2	90.8	74.5	**90.4**	91.0

Table 3.3: Quantitative results of different approaches for the semantic segmentation task on the Stanford background dataset [1]. We show the accuracy for each semantic class, the average accuracy of these eight classes (MCA), and the overall pixel accuracy (overall).

	sky	horizontal	vertical	average	overall
ConvNet	90.1	92.8	93.2	92.0	93.5
ConvNet-SVM	90.5	94.2	93.8	92.8	94.0
ConvNet-CSVM	90.6	93.9	94.6	93.1	93.8
ConvNet-CSVM2	90.5	95.3	96.1	**94**	95.1
ConvNet-WSVM	90.6	94.7	96.6	**94**	**95.2**

(a)

Method	overall	average
Gould et al. [1]	89.1	-
ConvNet	93.5	92.0
ConvNet-SVM	94.0	92.8
ConvNet-CSVM	93.8	93.1
ConvNet-CSVM2	95.1	**94.0**
ConvNet-WSVM	**95.2**	**94.0**

(b)

Method	overall (PPA)	average (MCA)
Gould et al. [1]	76.4	-
Pylon [57]	81.9	72.4
Multiscale Net [58]	81.4	76.0
TM-RCPN [59]	82.3	79.1
DeconvNet [60]	84.2	78.4
LSTM-RNN [61]	78.56	68.79
CNN-CRF [56]	83.5	76.9
Zoom-Out [22]	86.1	80.9
ConvNet	90.4	89.1
ConvNet-SVM	90.8	89.3
ConvNet-CSVM	91.2	89.6
ConvNet-CSVM2	**91.6**	87.8
ConvNet-CSVM3	90.9	**90.4**
ConvNet-WSVM	91.0	**90.4**

(c)

Table 3.4: (a) Quantitative results of different approaches for the geometry labeling task on the Stanford background dataset [1]. (b) Comparison with Gould et al. [1] on the geometric labeling task on the Stanford background dataset. (c) Comparison with state-of-the-art semantic segmentation approaches on Stanford background (semantic) dataset [1].

are shown in Fig. 3.3. Some qualitative examples for geometric labeling are shown in Fig. 3.4.

We compare our approach with several baseline methods. The comparisons are summarized in Table 3.4 (b) and Table 3.4 (c). We can see that our proposed approach

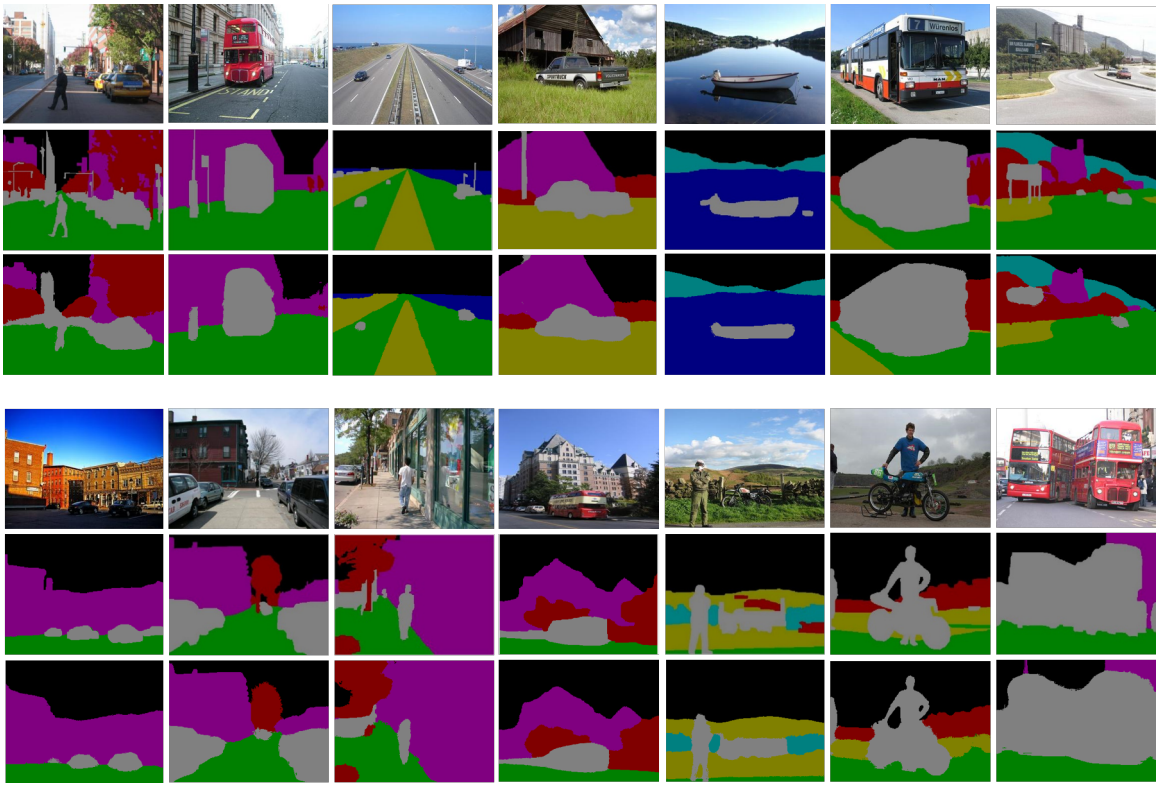


Figure 3.3: Sample results of semantic segmentation on the Stanford background dataset [1]. 1st row: test images; 2nd row: ground-truth semantic segmentations; 3rd-row: segmentation results produced by ConvNet-CSVM2. Different semantic classes are represented by different colors.

Figure 3.4: Sample results of geometric labeling on the Stanford background dataset [1]. 1st row: test images; 2nd row: ground-truth geometric labels; 3rd-row: geometric labeling results produced by ConvNet-CSVM2. Different geometric classes are represented by different colors.

	background	aeroplane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	**IoU**
FCN-8s [5]	91.2	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1	62.2
Zoom-out [22]	89.8	81.9	35.1	78.2	57.4	56.5	80.5	74.0	79.8	22.4	69.6	53.7	74.0	76.0	76.6	68.8	44.3	70.2	40.2	68.9	55.3	64.4
DeepLab-CRF [11]	93.1	84.4	54.5	81.5	63.6	65.9	85.1	79.1	83.4	30.7	74.1	59.8	79.0	76.1	83.2	80.8	59.7	82.2	50.4	73.1	63.7	71.6
DeConvNet+CRF [8]	92.9	87.8	41.9	80.6	63.9	67.3	88.1	78.4	81.3	25.9	73.7	61.2	72.0	77.0	79.9	78.7	59.5	78.3	55.0	75.2	61.5	70.5
CRFasRNN [25]	-	90.4	55.3	88.7	68.4	69.8	88.3	82.4	85.1	32.6	78.5	64.4	79.6	81.9	86.4	81.8	58.6	82.4	53.5	77.4	70.1	74.7
ConvNet	89.5	72.1	29.9	73.5	56.7	64.3	81.1	73.9	77.4	27.2	62.0	49.6	70.8	61.3	66.8	75.8	42.3	66.3	41.5	73.3	49.7	62.1
ConvNet-CSVM	83.0	79.2	30.1	77.5	54.3	67.4	80.8	75.4	76.0	29.6	62.3	53.2	68.5	63.1	68.1	75.4	46.2	69.7	40.8	73.8	52.6	63.2
ConvNet-CSVM2	86.2	77.5	29.4	78.1	54	66.9	83.7	77.1	76.7	32.8	63.2	52.9	73.2	63.4	70.4	77.5	44.6	70.1	40.8	53.1	74.3	64.1
ConvNet-WSVM	87.0	77.7	29.5	78.0	57	67.1	83.7	77.0	78.3	32.9	62.9	53.0	73.5	63.2	70.9	77.4	45.8	70.2	40.2	54.7	74.4	64.5

Table 3.5: Quantitative results of different approaches for the semantic segmentation task on the PASCAL VOC 2012 dataset [2].

outperforms all the baseline methods.

3.4.2 Evaluation result on PASCAL VOC 2012

The segmentation results on PASCAL VOC 2012 test set for different configurations are reported in Table 3.5. Following [5; 11; 25], we have used augmented training data with extra annotations for training the deep network. However, for training the SVM model, we didn't use any images other than the PASCAL VOC 2012 training set. Initially we achieve performance of 63.2% mean IoU for ConvNet-SVM and 64.5% mean IoU for Convnet-WSVM. Sample segmentation outputs are illustrated in Fig. 3.5. It is important to note that there is evidently an advantage in making use of the data and labels from the SBD that are not directly related to the PASCAL VOC 2012 problem, and this suggests value in the proposed approach in a more general sense given future availability of datasets that include dense pixel-wise labeling.

Figure 3.5: Sample results of semantic segmentation on the PASCAL VOC 2012 dataset [2]. 1st column: test images; 2nd column: ground-truth semantic segmentations; 3rd-column: segmentation results produced by ConvNet-CSVM2. Different semantic classes are represented by different colors.

Chapter 4

Gated Feedback Refinement for Coarse-to-Fine Dense Image Labeling

In this chapter, we propose a new approach for dense image labeling problems (e.g. semantic segmentation). For example, consider Fig. 4.1 that has fine objects such as a column-pole, pedestrian or bicyclist which require extraction of very fine details in order to be segmented accurately. Naturally, ConvNets try to extract exactly this type of fine-grained high spatial-frequency variation in the early convolution stages, even though corresponding object specific representations may not emerge until very late in the feed-forward ConvNet architecture. A question that naturally follows from this is: How can we incorporate these fine details in the segmentation process to get precise labeling while also carrying a rich feature-level representation?

At the deepest stage of encoding, one has the richest possible feature represen-

Figure 4.1: Labeling objects such as the column-pole, pedestrian or bicyclist shown above requires low-level finer details as well as high-level contextual information, despite of varying light scenes. In this work, we propose a gated feedback refinement network, which can be used with any bottom-up, feed-forward ConvNet. We show that the finer features learnt by our approach lead to significantly improved semantic segmentation.

tation, and relatively poor spatial resolution from a per-neuron perspective. While spatial resolution may be poor from a per-neuron perspective, this does not necessarily imply that recovery of precise spatial information is impossible. For example, a coarse coding strategy [62; 63] may allow for a high degree of precision in spatial localization but at the expense of the diversity of features encoded and involved in discrimination. An important implication of this, is that provided the highest layer does not require the power to precisely localize patterns, a much richer feature level representation is possible.

Information carried among earlier layers of encoding do have greater spatial locality, but may be less discriminative. Given that there is an extant representation of image characteristics at every layer, it is natural to assume that value may be had in leveraging earlier encoding representations at the decoding stage. In this manner, spatial precision that may be lost at deep layers in encoding may be gradually recovered from earlier representations. This removes some of the onus on deeper layers to

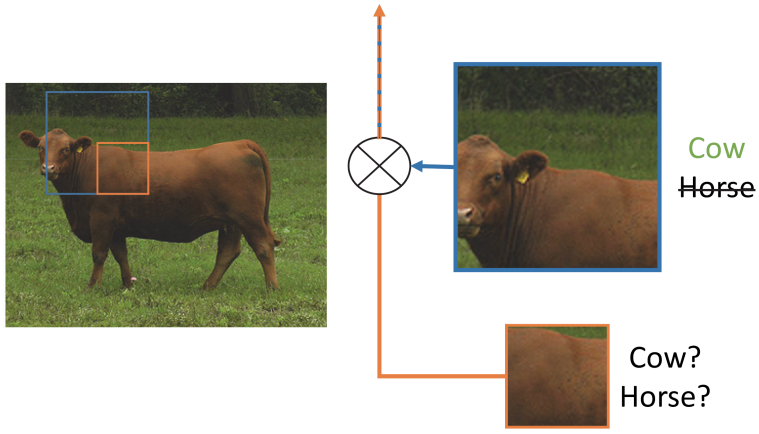


Figure 4.2: An illustration of the relationship between receptive field size across layers, and ambiguity that may arise. In this case, the larger (and more discriminative) receptive field (blue) resides at a deeper layer of the network, and may be of value in refining the representation carried by an earlier layer (orange) to resolve ambiguity and improve upon labeling performance.

represent highly discriminative characteristics of the image, while simultaneously facilitating precise localization. This intuition appears in the model we propose, as seen in connections between encoder layers and decoder layers in our network. This implies the shift in responsibility among encoding layers, and the associated discriminative power or capacity deeper in the network.

If one were to label categories within the image using only early layers, this may be problematic, especially in instances where local parts are ambiguous. The re-use of information from earlier encoder layers at the decoding stage is weakened by their lack of discrimination. For example, if one assumes reliance on convolution, and unpooling (which involves a fixed set of weights) to recover information and ultimately assign labels, this implies that any ambiguous representations are necessarily involved in decoding, which may degrade the quality of predictions. For example, while a convolutional layer deep within the network may provide strong discrimination between

a cow and a horse, representations from earlier layers may be specific to animals, but express confidence for both. If this confidence is passed on to the decoding stage, and a fixed scheme for combining these representations is present, this contributes to error in labeling. This observation forms the motivation for the most novel and important aspect of our proposed model and this intuition is illustrated in Fig. 4.2. While information from early encoding layers may be of significant value to localization, it is sensible to filter this information such that categorical ambiguity is reduced. Moreover, it is natural to use deeper, more discriminative layers in filtering information passed on from less discriminative, but more finely localized earlier layers.

The precise scheme that achieves this is discussed in detail in the remainder of this chapter. We demonstrate that a high degree of success may be achieved across a variety of benchmarks, using a relatively simple model structure in applying a canonical gating mechanism that may be applied to any network comprised of encoder and decoder components. This is also an area in which parallels may be drawn to neural information processing in humans, wherein more precisely localized representations that may be ambiguous are modulated or gated by higher-level features, iteratively and in a top-down fashion [42].

Based on the above observations, it is evident that features from all levels of a neural hierarchy (high+low) are of value, or even necessary for accurate dense image labeling. High-level or global features usually help in recognizing the category label while low-level or local features can help in assigning precise boundaries to the objects since spatial details tend to get lost in high-level/global features. In this thesis, we propose a novel end-to-end network architecture that effectively exploits multi-level

features in semantic segmentation.

4.1 Background

Encoder-Decoder Architecture: Our model (Fig. 4.4) is based on a deep encoder-decoder architecture (e.g. [13; 8]) used for dense image labeling problems such as semantic segmentation. The encoder network extracts features from an image and the decoder network produces semantic segmentation from the features generated by the encoder network. The encoder network is typically a CNN with alternating layers of convolution, pooling, non-linear activation, etc. The output of each convolution layer in the encoder network can be interpreted as features with different receptive fields. Due to spatial pooling, the spatial dimensions of the feature map produced by the encoder network are smaller than the original image. The subsampling layers in CNNs allow the networks to extract high-level features that are translation invariant, which are crucial for image classification. However, they reduce the spatial extent of the feature map at each layer in the network. Let $I \in \mathbb{R}^{h \times w \times d}$ be the input image (where h, w are spatial dimensions d is the color channel dimension) and $f(I) \in \mathbb{R}^{h' \times w' \times d'}$ be the feature map volume at the end of the encoder network. The feature map $f(I)$ has smaller spatial dimensions than the original image, i.e. $h' < h$ and $w' < w$. In order to produce full-sized segmentation results as the output, an associated decoder network is applied to the output of the encoder network to produce output that matches the spatial size of the original image. The fundamental differences between our work and various research contributions in prior work mainly lie in choices of the structure and composition of the decoder network. The decoder

in SegNet [13] progressively enlarges the feature map using an upsampling technique without learnable parameters. The decoder in DeconvNet [8] is similar, but has learnable parameters. FCN [5] uses a single layer interpolation for deconvolution. The decoder network in general enlarges the feature map using upsampling and unpooling in order to produce the final semantic segmentation result. Many popular CNN-based semantic segmentation models fall into this encoder-decoder framework, e.g. FCN [5], SegNet [13], DeconvNet [8].

Skip Connections: In a standard encoder-decoder architecture, the feature map from the top layer of the encoder network is used as the input for the decoder network. This feature map contains high-level features that tend to be invariant to "nuisance factors" such as small translation, illumination, etc. This invariance is crucial for certain high-level tasks such as object recognition, but is not ideal for many dense image labeling tasks (e.g. semantic segmentation) that require precise pixel-wise information, since important relationships may be abstracted away. One possible solution is to use "skip connections" [21; 5]. A skip connection directly links an encoder layer to a decoder layer. Since the bottom layers in the encoder network tend to contain precise pixel-wise information, the skip connections allow this information to be directly passed to the decoder network to produce the final segmentation result. In this section, we describe background most relevant to our proposed model in this chapter. In the following sections, we firstly describe our base network called Label Refinement Network (LRN) (Sec. 4.2). Then we discuss our final network Gated Feedback Refinement Network (G-FRNet) (Sec. 4.3) which builds on LRN. Fig. 4.3 illustrates the basic difference between these two network architectures.

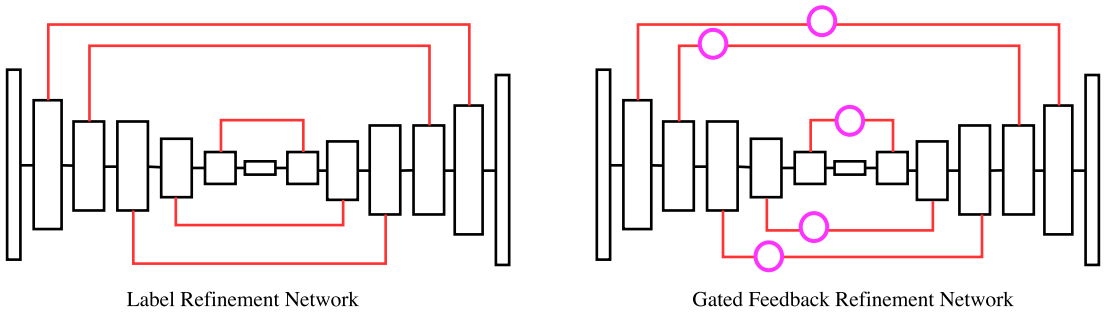


Figure 4.3: Overview of our Label Refinement network and Gated Feedback Refinement Network (G-FRNet). In LRN, each of the decoder layers has long range connections to its corresponding encoder layer whereas G-FRNet adds a gate between the long range connections to modulate the influence of these connections.

4.2 Label Refinement Network

In this section, we describe a novel network architecture called the *Label Refinement Network (LRN)* for semantic segmentation. The architecture of LRN is the simpler version of the one shown in Fig. 4.4, where the difference is that there are no gate units in LRN. Similar to prior work [13; 5; 8], LRN also uses an encoder-decoder framework. The encoder network of LRN is similar to that of SegNet [13] which is based on the VGG16 network [17]. The novelty of LRN lies in the decoder network. Instead of making the prediction at the end of the network, the decoder network in LRN makes predictions in a coarse-to-fine fashion at several stages. In addition, LRN also has deep supervision which is aimed to provide supervision early in the network by adding loss functions at multiple stages (not just at the last layer) of the decoder network.

The LRN architecture is motivated by the following observations: Due to the convolution and subsampling operations, the feature map $f(I)$ obtained at the end of the encoder network mainly contains high-level information about the image (e.g.

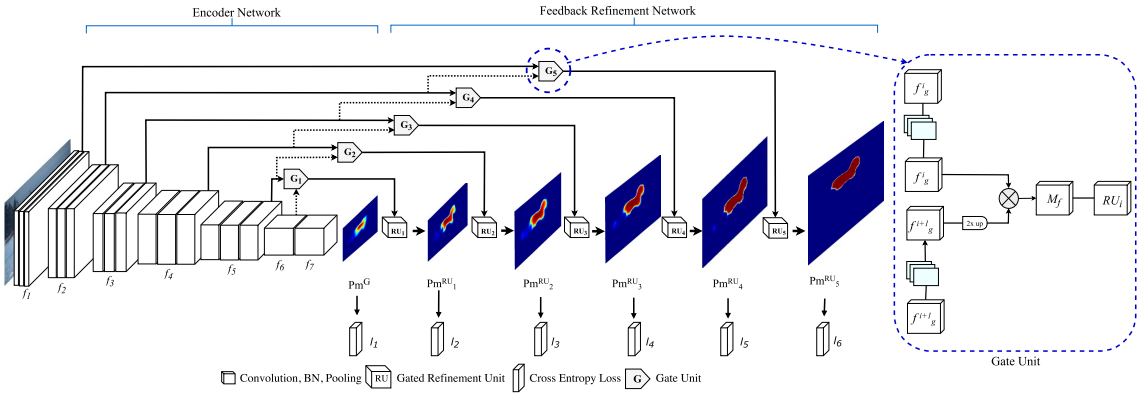


Figure 4.4: Overview of our Gated Feedback Refinement Network (G-FRNet). We use feature maps with different spatial dimensions produced by the encoder (f_1, f_2,..., f_7) to reconstruct a small (i.e. coarse) label map Pm^G. The decoder progressively refines the label map by adding details from feature maps in the encoder network. At each stage of decoding, a refinement unit (RU_1, RU_2,..., RU_5) produces a new label map with larger spatial dimensions by taking information from the previous label map and encoder layers as inputs (denoted by the edge connecting G_i and RU_i). The main novelty of the model is that information from earlier encoder layers passes through a gate unit before being forwarded to the decoder. We use standard 2x bilinear upsampling on each class score map before passing it to the next stage refinement module. We also use down-sampled ground-truth label maps to provide supervision (l_1, l_2, ..., l_6) at each decoding stage.

objects). Spatially precise information is lost in the encoder network, and therefore $f(I)$ cannot be used directly to recover a full-sized semantic segmentation which requires pixel-precision information. However, $f(I)$ contains enough information to produce a *coarse* semantic segmentation. In particular, we can use $f(I)$ to produce a segmentation map $s(I)$ of spatial dimensions $h' \times w'$, which is smaller than the original image dimensions $h \times w$. Our decoder network then progressively refines the segmentation map $s(I)$. Let $s_k(I)$ be the k-th segmentation map (left to right in Fig. 4.4). If $k > k'$, the segmentation map $s_k(I)$ will have a larger spatial dimensions than $s_{k'}(I)$. Note that one can interpret $s_k(I)$ as the feature map in the decoder network in most work (e.g. [13; 5; 8]). However, our model enforces the channel

dimension of $s_k(I)$ to be the same as the number of class labels, so $s_k(I)$ can be considered as a (soft) label map.

The network architecture in Fig. 4.4 has 7 convolution layers in the encoder. We use $f_k(I) \in \mathbb{R}^{h_k \times w_k \times d_k}$ $(k = 1, 2, ..., 7)$ to denote the feature map after the k-th convolution layer. The decoder network generates 6 label maps s_k $(k = 1, 2, ..., 6)$. After the last convolution layer of the encoder network, we use a 3×3 convolution layer to convert the convolution feature map $f_7(I) \in \mathbb{R}^{h_7 \times w_7 \times C}$ to $s_1(I) \in \mathbb{R}^{h_7 \times w_7 \times C}$, where C is the number of class labels. We then define a loss function on $s_1(I)$ as follows. Let $Y \in \mathbb{R}^{h \times w \times C}$ be the ground-truth segmentation map, where the label on each pixel is represented as a C-dimensional vector using the one-shot representation. We use $R_1(Y) \in \mathbb{R}^{h_7 \times w_7 \times C_7}$ to denote the segmentation map obtained by resizing Y to have the same spatial dimensions as $s_1(I)$. We can then define a loss function ℓ_1 (cross entropy loss is used) to measure the difference between the resized ground-truth $R_1(Y)$ and the coarse prediction $s_1(I)$ (after softmax). In other words, these operations can be written as:

$$\begin{aligned} s_1(I) &= \text{conv}_{3\times3}\big(f_7(I)\big) \\ \ell_1 &= \text{Loss}\Big(R_1(Y), \text{softmax}\big(s_1(I)\big)\Big) \end{aligned} \tag{4.1}$$

where $\text{conv}_{3\times3}(\cdot)$ denotes the 3×3 convolution and $\text{Loss}(\cdot)$ denotes the cross entropy loss function.

Now we explain how to get the subsequent segmentation map $s_k(I)$ $(k > 1)$ with larger spatial dimensions. One simple solution is to upsample the previous segmentation map $s_{k-1}(I)$. However, this upsampled segmentation map will be very coarse. In order to derive a more precise segmentation, we use the idea of skip-

connections [21; 5]. The basic idea is to make use of the outputs from one of the convolutional feature layers in the encoder network. Since the convolutional feature layer contains more precise spatial information, we can combine it with the upsampled segmentation map to obtain a refined segmentation map (see Fig. 4.5 for detailed illustration of our refinement module). In our model, we simply concatenate the outputs from the skip layer with the upsampled decoder layer to create a larger feature map, then use 3×3 convolution across the channels to convert the channel dimension to C. For example, the label map $s_2(I)$ is obtained from upsampled $s_1(I)$ and the convolutional feature map $f_5(I)$ (see Fig. 4.4 for details on these skip connections). We can then define a loss function on this (larger) segmentation map by comparing $s_k(I)$ with the resized ground-truth segmentation map $R_k(Y)$ of the corresponding size. These operations can be summarized as follows:

$$s_k(I) = \text{conv}_{3\times3}\Big(\text{concat}\Big(\text{upsample}\big(s_k(I)\big), f_{7-k}(I)\Big)\Big) \tag{4.2}$$

$$\ell_k = \text{Loss}\Big(R_k(Y), \text{softmax}\big(s_k(I)\big)\Big), \quad \text{where } k = 2, .., 6 \tag{4.3}$$

4.3 Gated Feedback Refinement Network

In this section, we describe our proposed network called *Gated Feedback Refinement Network* (G-FRNet) for semantic segmentation that is inspired by LRN.

G-FRNet is built upon LRN. LRN uses skip connections to directly connect two layers in a network, i.e. an encoder layer to an decoder layer. For example, in the network architecture of Fig. 4.4, a traditional skip connection might connect f_5 with Pm^{RU_1}. Although this allows the network to pass finer detailed information from the

early encoder layers to the decoder, it may degrade the quality of predictions. As mentioned earlier, the categorical ambiguity in early encoder layers may be passed to the decoder.

The main novelty G-FRNet is that we use a gating mechanism to modulate the information being passed via the skip connections. For example, say we want to have a skip connection to pass information from the encoder layer f_5 to the decoder layer Pm^{RU_1}. Instead of directly passing the feature map f_5, we first compute a gated feature map based on f_5 and an encoder layer above (i.e. f_6 in Fig. 4.4). The intuition is that f_6 contains information that can help resolve ambiguity present in f_5. For instance, some of the neurons in f_6 might fire on image patches that look like an animal (either cow or horse). This ambiguity about categories (cow vs. horse) cannot be resolved by f_5 alone since the receptive field corresponding to this encoder layer might not be large or discriminative enough. But the encoder layer (e.g. f_6) above may not be subject to these limitations and provide unambiguous confidence for the correct category. By computing the gated feature map from f_5 and f_6, categorical ambiguity can be filtered out before reaching the decoding stage where spatial precision is recovered. Fig. 4.2 provides an example of categorical ambiguity.

The gated feature map from G_1 contains information about finer image details. We then combine it with the coarse label map Pm^{G} to produce an enlarged label map Pm^{RU_1}. We repeat this process to produce progressively larger label maps ($\mathrm{Pm}^{\mathrm{RU}_1}$, $\mathrm{Pm}^{\mathrm{RU}_2}$, $\mathrm{Pm}^{\mathrm{RU}_3}$, $\mathrm{Pm}^{\mathrm{RU}_4}$, $\mathrm{Pm}^{\mathrm{RU}_5}$).

In the following sections we discuss Gate Unit (Sec. 4.3.1), Gated Refinement Unit (Sec. 4.3.2) and Stage-wise supervision in detail (Sec. 4.3.3).

4.3.1 Gate Unit

Previous work [36] proposed refinement across different levels by combining convolution features from earlier layers. Instead of combining convolution features with coarse label maps directly, we introduce gate units to control the information passed on. The gate units are designed to control the information passed on by modulating the response of encoder layers for each spatial region in a top-down manner. Fig. 4.4 (right) illustrates the architecture of a gate unit.

The gate unit takes two consecutive feature maps f_g^i and f_g^{i+1} as its input. The features in f_g^i are high-resolution with smaller receptive fields (i.e. small context), whereas features in f_g^{i+1} are of low-resolution with larger receptive fields (i.e. large context). A gate unit combines f_g^i and f_g^{i+1} to generate rich contextual information. Alternative approaches apply a refinement process straight away that combines convolution features (using skip connections [5]) with coarse label maps through concatenation to generate a new label map. In this case, it is less likely that the model will take full advantage of the contribution of higher resolution feature maps if they carry activation that is ambiguous with respect to class. As a result, skip connections alone have inherent limitations in discerning missing spatial details. Therefore, unlike skip connections we first obtain a gated feature map before passing on the higher resolution encoding to the refinement unit. As a result, contextual features will be assigned higher gate values and retain their activation for each successive stage of refinement, while irrelevant/noise regions will be suppressed.

We now explain how we obtain a gated feature map from a gate unit. Given that the dimensions of feature map f_g^i and the gate control input f_g^{i+1} might not be the

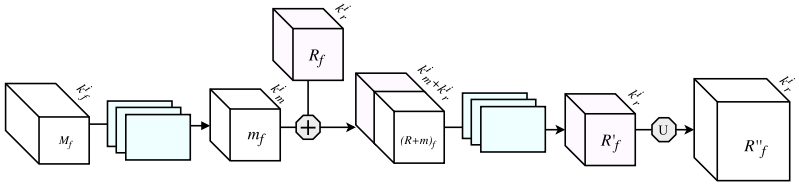


Figure 4.5: Detailed overview of a Gated Refinement Unit. The refinement unit is unfolded here for i^{th} stage. The refinement module (similar to [3]) is composed of convolution, batch normalization, concatenation, and upsampling operations.

same, we first use a transformation function $T_f : \Re^{\varepsilon} \mapsto \Re^{\chi}$ to map f_g^{i+1} to $f_{g\prime}^{i+1}$. In the tranformation process, a sequence of operations is carried out on f_g^i and f_g^{i+1} followed by a element-wise product. Firstly, we apply a 3×3 convolution with batch normalization and ReLU to both feature maps. After these operations, let c_g^i and c_g^{i+1} be the number of channels in f_g^i and f_g^{i+1} such that $c_g^i = c_g^{i+1}$. f_g^{i+1} is then upsampled by a factor of 2 to produce a new feature map $f_{g\prime}^{i+1}$ whose spatial dimensions match f_g^i. We obtain the i^{th} stage gated (from gate G_i in Fig. 4.4) feature map M_f from the element-wise product between f_g^i and $f_{g\prime}^{i+1}$. Finally, the resultant feature map M_f is fed to the gated refinement unit (see Sec. 4.3.2). The formulation of obtaining a gated feature map M_f from gate unit G_i can be written as follows:

$$v_i = T_f(f_g^{i+1}), u_i = T_f(f_g^i), M_f = v_i \odot u_i \tag{4.4}$$

where $\odot$ denotes element-wise product. As gate units are integrated with the end-to-end network, the parameters could be trained with the back-propagation (BP) algorithm.

4.3.2 Gated Refinement Unit

Fig. 4.5 shows in detail the architecture of our gated refinement unit (see RU in Fig. 4.4). Each refinement unit RU^i takes a coarse label map R_f with channel k_r^i (generated at $(i-1)^{th}$ stage of the FRN) and gated feature map M_f as its input. RUs learn to aggregate information and generate a new label map R'_f with larger spatial dimensions through the following sequence of operations: First, we apply a 3×3 convolution followed by a batch normalization layer on M_f to obtain a feature map m_f with channel k_m^i. In our model configuration, $k_m^i = k_r^i = C$ where C is the number of possible labels. Next, m_f is concatenated with the prior stage label map R_f, producing feature map $(R+m)_f$ with $k_m^i + k_r^i$ channels. There are two reasons behind making $k_m^i = k_r^i$. First, the channel dimension of the feature map obtained from the encoder is typically very large (i.e. $c_g^i \gg k_r^i$). So directly concatenating R_f with a feature map containing a larger number of channels is computationally expensive. Second, concatenating two feature maps having a large difference in the number of channels risks dropping signals from the representation with fewer layers. Finally, the refined label map R'_f is generated by applying a 3×3 convolution. Note that R'_f is the i^{th} stage prediction map. The prediction map R'_f is upsampled by a factor of 2 and fed to the next stage $(i+1)^{th}$ gated refinement unit. These operations can be summarized as follows:

$$m_f = \mathbb{B}(\mathbb{C}_{3\times3}(M_f)), \gamma = m_f \oplus R_f, R'_f = \mathbb{C}_{3\times3}(\gamma) \tag{4.5}$$

where $\mathbb{B}(.)$, $\mathbb{C}(.)$, and $\oplus$ refer to batch normalization, convolution, and concatenation respectively.

4.3.3 Stage-wise Supervision

Our network produces a sequence of label maps with increasing spatial dimensions at the decoder stage, although we are principally interested in the label map at the last stage of the decoding. Label maps produced at earlier stages of decoding might provide useful information as well and allow for supervision earlier in the network. Following [3], we adopt the idea of deep supervision [32] in our network to provide stage-wise supervision on predicted dense label maps. In more specific terms, let $I \in \mathbb{R}^{h \times w \times d}$ be a training sample with ground-truth mask $\eta \in \mathbb{R}^{h \times w}$. We obtain k resized ground-truth maps $(R_1, R_2, ..., R_k)$ by resizing η. We define a loss function l_i (pixel-wise cross entropy loss is used) to measure the difference between the resized ground-truth $R_i(\eta)$ and the predicted label map at each stage of decoding. We can write these operations as follows:

$$l_k = \begin{cases} \xi\Big(R_i(\eta), Pm^{G}\Big) & \text{if} \quad \text{i} = 1 \\ \xi\Big(R_i(\eta), Pm^{RU_i}\Big) & \text{otherwise} \end{cases} \tag{4.6}$$

where ξ denotes weighted cross-entropy loss which is defined by the following:

$$\xi = -\lambda \sum_{t=1}^{N} \sum_{t=1}^{C} t_i \log(x_i), \qquad \ell = \sum_{k=1}^{6} l_k \tag{4.7}$$

λ is the class balancing frequency or stage-specific weight factor on a per-pixel term basis. The loss function ℓ in our network is the summation of cross-entropy losses at various stages of refinement network. The network is trained using back-propagation to optimize this loss. Fig. 4.6 illustrates the effectiveness of the gated refinement scheme. We can see that the refinement scheme progressively improves the spatial details of dense label maps. It also shows that the top convolution layer (conv7 in

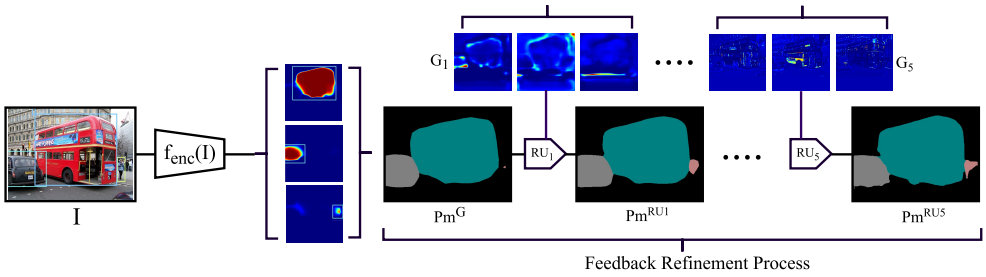


Figure 4.6: Visualization of hierarchical gated refinement scheme. The refinement process integrates higher-frequency details with the lower resolution label map at each stage. Class-wise activation maps for each gate are shown as heatmaps.

our encoder network) can predict a coarse label map without capturing finer image details. The feedback refinement network is able to recover missing details (e.g. the boundaries of the bus and the car) in the coarse label map.

4.4 Experiments

In this section, we first provide implementation details (Sec. 4.4.1). Then we present experimental results on five challenging dense labeling benchmark datasets: Cambridge Driving Labeled Video (CamVid) (Sec. 4.4.2), PASCAL VOC 2012 (Sec. 4.4.3), Horse-Cow Parsing (Sec. 4.4.4), PASCAL-Person-Part dataset (Sec. 4.4.5), and SUN-RGBD dataset (Sec. 4.4.6).

4.4.1 Implementation Details

We have implemented our network using Caffe [54] on a single Titan X GPU. Pre-trained VGG-16 [17] parameters are used to initialize the convolution layers in the encoder network (i.e. *conv*1 to *conv*5 layer). Other convolution layers' parameters are randomly assigned based on Xavier initialization. Randomly cropped patches of size

($h_{min} \times w_{min}$) are fed into the network. We set ($h_{min} \times w_{min}$) to 320×320 for Pascal VOC and 360×480 for CamVid and Horse-Cow parsing datasets. For the PASCAL VOC 2012 dataset, we normalize the data using VGG-16 mean and standard deviation. We employ pixel-wise cross entropy loss (with equal weights) as the objective function to be optimized for all the semantic categories. For the CamVid dataset, since data is not balanced we use weighted cross entropy loss following previous work [13]. The weights are computed using the class balancing technique proposed in [28].

Since the gated refinement modules can handle inputs of any size, we are able to test our network with original image size. The spatial resolution of the final segmentation map is therefore the same as the test image.

To demonstrate the merit of individual component of our model, we also perform an ablation study by comparing with the following variants of our proposed model:

LRN: Label Refinement Network that only uses upper layer features for refinement.

G-FRNet: Gated Feedback refinement network with gating mechanism. The major difference between G-FRNet and LRN is the gate units. To demonstrate the real impact of gate units we consider LRN as our base model and report experimental results for both models in all of our comparisons.

G-FRNet-Res101: Gated Feedback refinement network where the encoder/base model (VGG-16) is replaced with dilated ResNet-101 [24]. Inspired by [24], we use the "poly" learning rate policy defined by $(1 - \frac{iter}{maxiter})^{power}$ when training G-FRNet-Res101. The learning rate of the newly added layers are initialized as $2.5 \times 10^{-}3$ and that of other previously learned layers initialized as 2.5×10^{-4}. We train the

ResNet-101 based models using stochastic gradient descent with a batch size of 1, momentum of 0.9, and weight decay of 0.0005.

4.4.2 CamVid

Method	Building	Tree	Sky	Car	Sign-Symbol	Road	Pedestrian	Fence	Column-Pole	Sidewalk	Bicyclist	**Mean IoU**
	w/o ConvNet											
SuperParsing [4]	70.4	54.8	83.5	43.3	25.4	83.4	11.6	18.3	5.2	57.4	8.9	42.0
TextonBoost + FSO [4]	74.4	71.8	91.6	64.9	27.7	91.0	33.8	34.1	16.8	73.9	27.6	55.2
	with ConvNet											
Bayesian SegNet [78]	n/a											63.1
DeconvNet [8]	n/a											48.9
ReSeg Net [65]	n/a											58.8
SegNet [13]	68.7	52	87	58.5	13.4	86.2	25.3	17.9	16.0	60.5	24.8	50.2
FCN-8s [5]	77.8	71.0	88.7	76.1	32.7	91.2	41.7	24.4	19.9	72.7	31.0	57.0
DeepLab-LargeFOV [11]	81.5	74.6	89.0	82.2	42.3	92.2	48.4	27.2	14.3	75.4	50.1	61.6
Dilation [26]	82.6	76.2	89.9	84.0	46.9	92.2	56.3	35.8	23.4	75.3	55.5	65.3
Dilation + FSO + DiscreteFlow [4]	**84.0**	77.2	91.3	**85.6**	**49.9**	92.5	59.1	37.6	16.9	76.0	57.2	66.1
DenseNet103 [67]	83.0	**77.3**	**93.0**	77.3	43.9	**94.5**	**59.6**	37.1	**37.8**	82.2	50.5	66.9
LRN [3]	78.6	73.6	76.4	75.2	40.1	91.7	43.5	41.0	30.4	80.1	46.5	61.7
G-FRNet	82.5	76.8	92.1	81.8	43.0	**94.5**	54.6	**47.1**	33.4	**82.3**	**59.4**	**68.0**

Table 4.1: Quantitative results on the CamVid dataset [64]. We report per-class IoU and mean IoU for each method. We split methods into two categories depending on whether or not they use ConvNet. Not surprisingly, ConvNet-based methods typically outperform non-ConvNet methods. Our approach achieves state-of-the-art results on this dataset. Note that the improvements on smaller and finer objects are particularly pronounced for our model.

The Cambridge-driving Labeled Video (CamVid) dataset [64] consists of 701 high resolution video frames extracted from a video footage recorded in a challenging urban setting. Ground-truth labels are annotated according to one of 32 semantic categories. Following [4; 13; 65], we consider 11 larger semantic classes (road, building, sky, tree, sidewalk, car, column-pole, fence, pedestrian, bicyclist, and sign-symbol) for evaluation. We split the dataset into training, validation, and test sets following [66].

Finally, we have 367 training images, 100 validation images, and 233 test images. In order to make our experimental settings comparable to previous works [4; 26; 65; 13], we downsample the images in the dataset by a factor of 2 (i.e. 480×360). Table 4.1 shows the results of our model and comparisons with other state-of-the-art approaches on this dataset, demonstrating that we achieve state-of-the-art results on this dataset. G-FRNet has produced significantly better performance over other recently developed segmentation network architectures including SegNet [13], DilatedNet [26], FSO [4], DeepLab [11], DenseNet [67], etc. For each method, we report the category-wise IoU score and mean IoU score. LRN [3] outperforms SegNet [13] by more than 11% (in terms of mean IoU) while our approach (i.e. G-FRNet) achieves an accuracy gain of 6% when compared with DeepLab [11] and by almost 2% over Dilation [26] and FSO [4].

Fig. 4.7 shows some qualitative results on this dataset. We can see that our model is especially accurate for challenging object categories, such as column-pole, sidewalk, bicyclist, and sign-symbols compared to [4].

4.4.3 PASCAL VOC 2012

PASCAL VOC 2012 [2] is a challenging dataset for semantic segmentation. This dataset consists of 1,464 training images and 1,449 validation images of 20 object classes (plus the background class). There are 1,456 test images for which ground-truth labels are not publicly available. We obtained results on the test set by submitting our final predictions to the evaluation server. Following prior work [11; 5; 13], we augment the training set with extra labeled PASCAL VOC images from [71]. In the

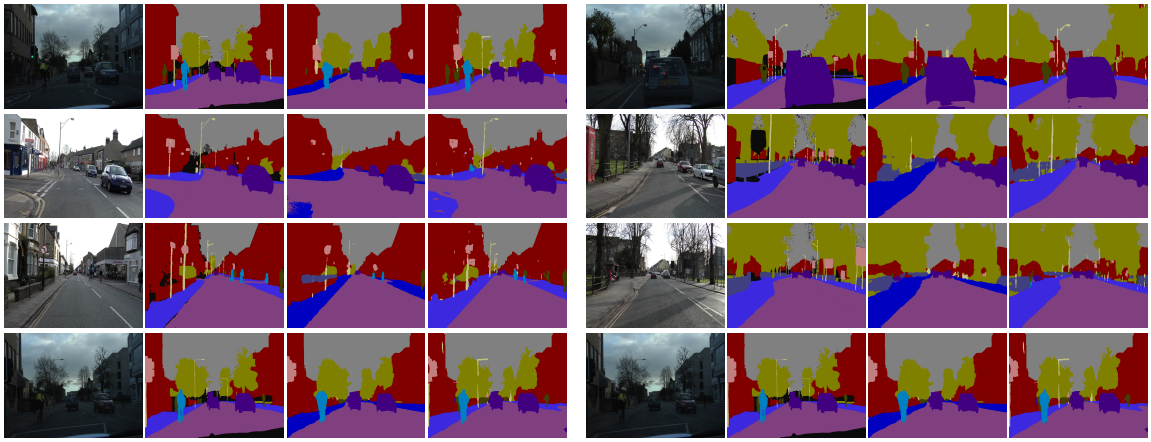


Figure 4.7: Qualitative results on the CamVid dataset. G-FRNet is capable of retaining the shape of smaller and finer object categories (e.g. column-pole, side-walk, bicyclist, and sign-symbols) accurately compared to FSO [4].

Method	Mean IoU (%)
DeepLab-CRF-LargeFOV [11]	67.6
DeepLab-MSc-CRF-LargeFOV [11]	68.7
FCN [5]	61.3
FCN + CRF [5]	63.7
OA-Seg + CRF [9]	70.3
DPN [68]	67.8
CRF-RNN [25]	69.6
DeconvNet [8]	67.1
Attention [69]	71.4
PDNs [70]	76.7
DeepLabv2 [24]	77.7
LRN [3]	62.8
G-FRNet	68.7
G-FRNet + CRF	71.0
G-FRNet-Res101 + CRF	**77.8**

Table 4.2: Comparison of different methods on PASCAL VOC 2012 validation set. Note that DeconvNet [8] result is taken from [9].

end, we have 10,582 labeled training images. In Table 4.2, we compare our results on the validation set with previous works. G-FRNet + CRF achieves best result with 71.0% mean IoU accuracy compared to other models based on an encoder-decoder based architecture ([8; 9; 5]). When we switch to a base model that exhibits

method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	**mIoU**
Hypercolumn [21]	68.4	27.2	68.2	47.6	61.7	76.9	72.1	71.1	24.3	59.3	44.8	62.7	59.4	73.5	70.6	52.0	63.0	38.1	60.0	54.1	59.2
FCN-8s [5]	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1	62.2
SegNet [13]	74.5	30.6	61.4	50.8	49.8	76.2	64.3	69.7	23.8	60.8	54.7	62.0	66.4	70.2	74.1	37.5	63.7	40.6	67.8	53.0	59.1
Zoom-Out [22]	85.6	37.3	83.2	62.5	66.0	85.1	80.7	84.9	27.2	73.2	57.5	78.1	79.2	81.1	77.1	53.6	74.0	49.2	71.7	63.3	64.4
DeconvNet[8]	87.8	41.9	80.6	63.9	67.3	88.1	78.4	81.3	25.9	73.7	61.2	72.0	77.0	79.9	78.7	59.5	78.3	55.0	75.2	61.5	70.5
DeepLab [11]	84.4	54.5	81.5	63.6	65.9	85.1	79.1	83.4	30.7	74.1	59.8	79.0	76.1	83.2	80.8	59.7	82.2	50.4	73.1	63.7	71.6
CRFasRNN [25]	87.5	39.0	79.7	64.2	68.3	87.6	80.8	84.4	30.4	78.2	60.4	80.5	77.8	83.1	80.6	59.5	82.8	47.8	78.3	67.1	72.0
DPN [68]	87.7	59.4	78.4	64.9	70.3	89.3	83.5	86.1	31.7	79.9	62.6	81.9	80.0	83.5	82.3	60.5	83.2	53.4	77.9	65.0	74.1
Dilation [26]	91.7	39.6	87.8	63.1	71.8	89.7	82.9	89.8	37.2	84.0	63.0	83.3	89.0	83.8	85.1	56.8	87.6	56.0	80.2	64.7	75.3
Attention [69]	**93.2**	41.7	88.0	61.7	74.9	92.9	84.5	90.4	33.0	82.8	63.2	84.5	85.0	87.2	85.7	60.5	87.7	57.8	84.3	68.2	76.3
LRR [35]	92.4	45.1	**94.6**	65.2	75.8	95.1	**89.1**	92.3	**39.0**	85.7	**70.4**	88.6	89.4	**88.6**	86.6	**65.8**	86.2	57.4	85.7	**77.3**	79.3
DeepLabv2 [24]	92.6	**60.4**	91.6	63.4	76.3	95.0	88.4	92.6	32.7	88.5	67.6	89.6	**92.1**	87.0	**87.4**	63.3	88.3	**60.0**	86.8	74.5	**79.7**
LRN [3]	79.3	37.5	79.7	47.7	58.3	76.5	76.1	78.5	21.9	67.7	47.6	71.2	69.1	82.1	77.5	46.8	70.1	40.3	71.5	57.4	64.2
G-FRNet	84.8	39.6	80.3	53.9	58.1	81.7	78.2	78.9	28.8	75.3	55.2	74.7	75.5	81.9	79.7	51.7	76.3	43.2	80.1	62.3	68.2
G-FRNet + CRF	87.7	42.9	85.4	51.6	61.0	82.9	81.7	81.6	29.1	79.3	56.1	77.6	78.6	84.6	81.6	52.8	79.0	45.0	82.1	64.1	70.4
G-FRNet-Res101	91.4	44.6	91.4	**69.2**	**78.2**	**95.4**	88.9	**93.3**	37.0	**89.7**	61.4	**90.0**	91.4	87.9	87.2	63.8	**89.4**	59.9	**87.0**	74.1	79.3

Table 4.3: Quantitative results in terms of mean IoU on PASCAL VOC 2012 test set. Note that G-FRNet-Res101 includes CRF.

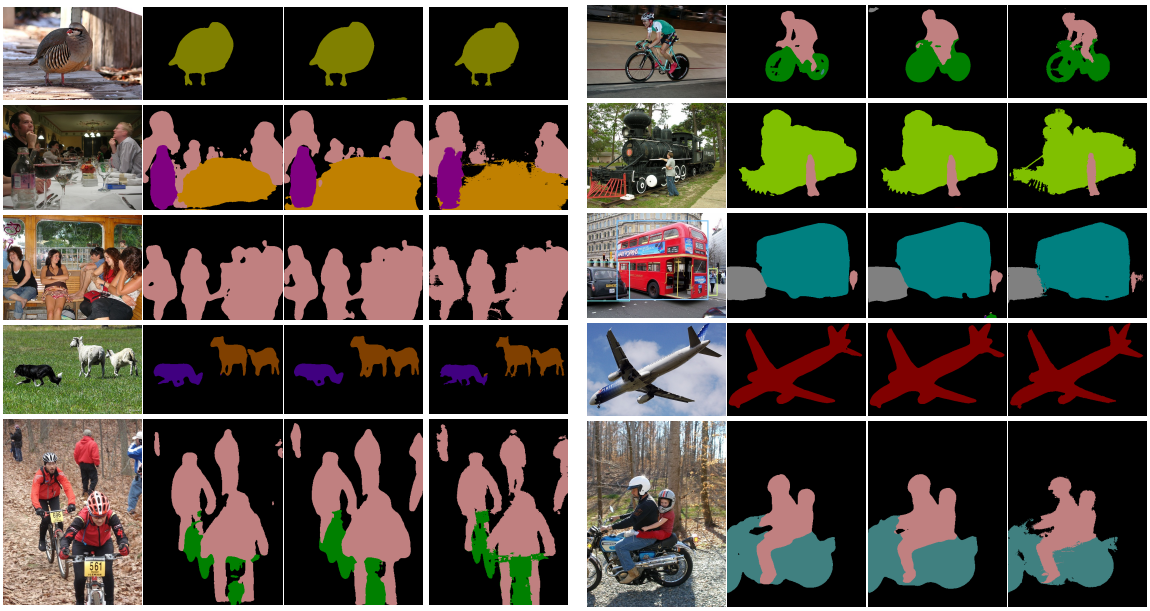


Figure 4.8: Qualitative comparisons between the FCN[5] model and our LRN model on the PASCAL VOC 2012 dataset.

stronger base performance (e.g. ResNet-101 [24] instead of VGG) our model G-FRNet-Res101 + CRF achieves 77.8% mean IoU which is very competitive compared to recent ResNet based state-of-the-art methods.

In order to evaluate on test set, we train our model on both the training and validation set. Table 4.3 shows quantitative results of our method on the test set. We achieve very competitive performance compared to other baselines. LRN [3] achieves 64.2% mean IoU which outperforms FCN [5] and SegNet [13]. Our proposed approach G-FRNet improves the mean IoU accuarcy by 4%. Many existing works (e.g. [11; 8; 24; 69]) use a CRF model [72] as a postprocessing to improve the performance. When we apply CRF on top of our final prediction (G-FRNet + CRF), we further improve the mean IoU to 70.4% on the test set. G-FRNet-Res101 (with CRF) further improves the performance and yields 79.3% mean IoU on test set which is very competitive compared to existing state-of-the-art approaches. A link to the results from the benchmark site is provided [1].

Fig. 4.8 shows qualitative results on the PASCAL VOC 2012 validation set. Overall, G-FRNet produces more accurate prediction maps compared to [5; 13; 11]. Note that our model is capable of handling multi-scale objects efficiently, whereas [5; 13; 11] fail in labeling larger and smaller scale objects due to fixed size receptive fields.

In recent years, many semantic segmentation methods have been proposed based on PASCAL VOC 2012 which are increasingly more precise in terms of IoU measure, and also introduce significant additional model complexity. However, there are only few recent methods [8; 13] that use a simpler encoder-decoder architecture for this problem, and it is most natural to compare our approach directly with this related family of models. Unlike other baseline methods, we obtain these results without employing any performance enhancing techniques, such as using object proposals [8] and multi-stage training [8]. It is worth noting that while the proposed model is shown

[1] http://host.robots.ox.ac.uk:8080/anonymous/HU5Y96.html

to be highly capable across several datasets, a deeper ambition of this approach is to demonstrate the power of basic information routing mechanisms provided by gating in improving performance. The encoder-decoder based architecture provides a natural vehicle for this demonstration. It is expected that a wide variety of networks that abstract away spatial precision in favor of a more complex pool of features may benefit from installing similar logic to the proposed gating mechanism.

4.4.4 Horse-Cow Parsing Dataset

	Horse						Cow					
Method	Bkg	head	body	leg	tail	IoU	Bkg	head	body	leg	tail	IoU
SPS [6]	79.14	47.64	69.74	38.85	-	-	78.0	40.55	61.65	36.32	-	-
SPS- Guidance [73]	76.0	55.0	52.4	46.8	37.2	50.3	69.7	57.6	62.7	38.5	11.8	48.03
HC [21]	85.71	57.30	77.88	51.93	37.10	61.98	81.86	55.18	72.75	42.03	11.04	52.57
JPO [74]	87.34	60.02	77.52	58.35	51.88	67.02	85.68	58.04	76.04	51.12	15.00	57.18
DeepLab-LargeFoV [11]	87.44	64.45	80.70	54.61	44.03	66.25	86.56	62.76	78.42	48.83	19.97	59.31
LG - LSTM [75]	89.64	**66.89**	84.20	60.88	42.06	68.73	89.71	68.43	82.47	53.93	19.41	62.79
LRN [3]	90.11	53.23	81.57	56.50	48.03	65.89	90.30	64.41	81.52	53.44	23.03	62.53
G-FRNet	**91.79**	60.44	**84.37**	**64.07**	**53.47**	**70.83**	**91.48**	**69.26**	**84.10**	**57.58**	**24.31**	**65.35**

Table 4.4: Comparison of object parsing performance with state-of-the-art methods on Horse-Cow parsing dataset [6].

To further confirm the value and generality of our model for dense labeling problems, we also evaluate our model on object parts parsing dataset introduced in [6]. This dataset contains images of horses and cows only, which are manually selected from the PASCAL VOC 2010 benchmark [2] based on most observable instances. The task is to label each pixel according to whether this pixel belongs to one of the body parts (head, leg, tail, body).

We split the dataset following [6] and obtain 294 training images and 227 test images. We compare the performance of our model with state-of-the art methods including the most recent method LG-LSTM [75].

Table 4.4 shows the performance of our models and comparisons with other baseline methods. LRN achieves competitive performance (65.89% and 62.53% mean IoU on horse and cow parsing respectively) whereas the proposed G-FRNet architecture makes further improvement and outperforms all the baselines in terms of mean IoU. The results reach 70.83% mean IoU for horse parsing and 65.35% for cow parsing. We also provide qualitative results in Fig. 4.9. The superior performance achieved by our model indicates that integrating gate units in the refinement process is very effective in capturing complex contextual patterns within images which play a critical role in distinguishing and segmenting different localized semantic parts of an instance.

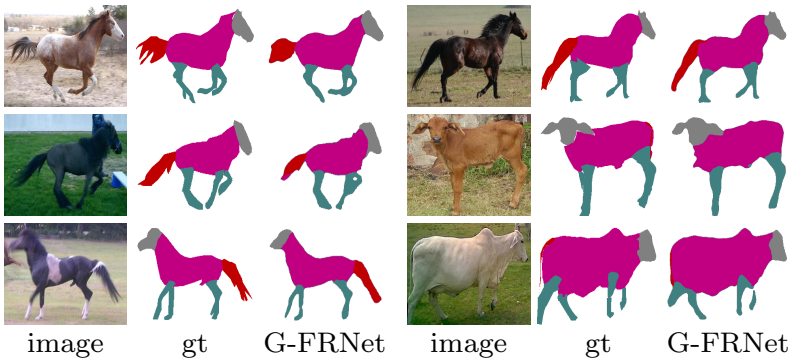


Figure 4.9: Qualitative results on Horse-Cow parsing dataset [6].

4.4.5 PASCAL-Person-Part

We further carry out experiments on the PASCAL-Person-Part dataset, a subset of PASCAL VOC 2010 images introduced in [6]. This dataset consists of humans images with variety of poses and scales. It includes pixel-level annotations for six person parts: Head, Torso, Upper/Lower Arms, Upper/Lower Legs and the Background. Following [24; 37], we use only those PASCAL VOC images that contains at least

one person. The dataset has 1717 training images and 1818 test images. We evaluate only our best network G-FRNet† on this dataset.

We report segmentation results of PASCAL-Person-Part dataset in Table 4.5 (a). We also compare our results with other state-of-the-art methods. The results clearly demonstrate the effectiveness of our network. Qualitative examples of our object parsing results on this dataset are shown in Table 4.5 (b)

Method	Mean IoU (%)
DeepLab [11]	51.8
LG-LSTM [75]	57.97
Attention [69]	56.39
HAZN [76]	57.54
Graph LSTM [77]	60.16
DeepLabv2 (ResNet-101) [24]	64.94
LRN-Res101 [3]	60.75
G-FRNet-Res101	64.61

(a)

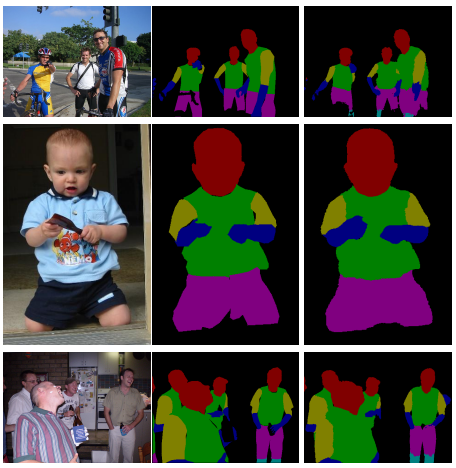


image ground-truth G-FRNet-Res101

(b)

Table 4.5: (a) Comparison of object parsing results with other state-of-the-art results on Pascal Person-Part dataset [6]. (b) Qualitative results on the Pascal Person-Part dataset.

4.4.6 SUN RGB-D

We also evaluate our model on the SUN RGB-D dataset [10] which contains 5,285 training and 5,050 test images. The images consist of indoor scenes of varying resolution and aspect ratio. There are 37 indoor scene classes with corresponding segmentation labels (background is not considered as a class and is ignored during training and testing). The segmentation labels are instance-wise, i.e. multiple instances of

Method	Pixel Acc.	Mean Acc.	Mean IoU
FCN-8s [5]	68.18	38.41	27.39
DeepLab [11]	71.90	42.21	32.08
DeconvNet [8]	66.13	33.28	22.57
SegNet [13]	70.3	35.6	26.3
Bayesian SegNet [78]	72.63	44.76	30.7
SegNet + DS	71.3	49.2	31.2
LRN [3]	72.5	46.8	33.1
G-FRNet-Res101	75.33	47.49	36.86

(a)

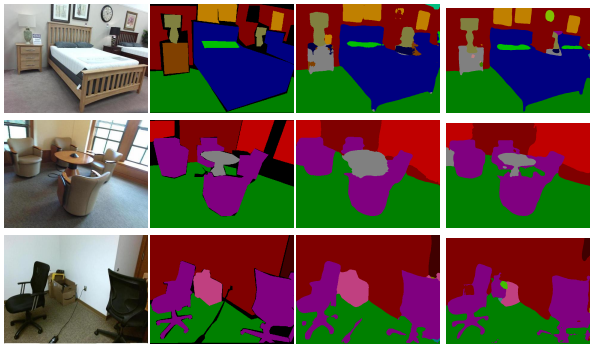


image ground-truth LRN G-FRNet-Res101

(b)

Table 4.6: (a) Comparison of scene parsing results with other state-of-the-art results on SUN RGB-D dataset [10]. (b) Qualitative results on the SUN RGB-D dataset.

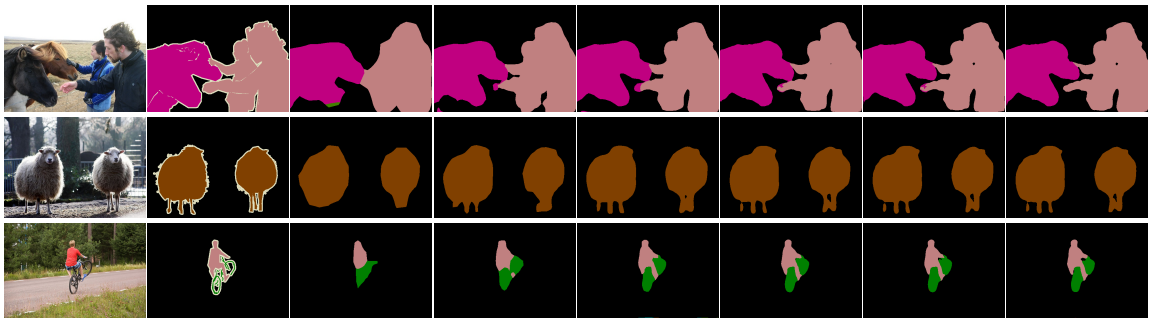


image ground-truth Sm^{G} Sm^{RU_1} Sm^{RU_2} Sm^{RU_3} Sm^{RU_4} Sm^{RU_5}

Figure 4.10: Stage-wise visualization of semantic segmentation results on PASCAL VOC 2012. For each row, we show the input image, ground-truth, and the prediction map produced at each stage of our feedback refinement network.

same class in an image have different labels. We convert the ground-truth labels into class-wise segmentation labels so that all instances of the same class have the same corresponding label. Although the dataset also contains depth information, we only use the RGB images to train and test our model. Quantitative results on this dataset are shown in Table 4.6 (a). Our LRN model achieves better mean IoU than other baselines on this dataset. G-FRNet-Res101 further improves the performance and yields 36.86% mean IoU. We show some qualitative results in Table 4.6 (b).

4.4.7 Ablation Analysis

To further illustrate the impact of the gated coarse-to-fine refinement, we show segmentation maps produced at different stages (see Fig. 4.10) in the network. We can see that gated coarse-to-fine refinement scheme progressively improves the details of predicted segmentation maps by recovering the missing parts (e.g., the leg of the person and sheep in the top and 3^{rd} row respectively).

We perform ablation analysis to demonstrate the benefit of our coarse-to-fine approach and gate units. We first perform a controlled study to isolate the effect of gate units on labeling accuracy. Then we include the gate units and train the model on three different datasets. Table 4.7 (a) and Table 4.7 (b) show the results of this stage-wise analysis on the datasets used. We can see that, from Pm^{G} to Pm^{RU_5}, mean IoU is progressively enhanced in all the datasets. Note that $\text{Pm}^{\text{G}}, .., \text{Pm}^{\text{RU}_5}$ are not results of separate stage-wise training. They are obtained from different stages of the feedback refinement network. The result of Pm^{G} is implicitly affected by the supervisions provided at $\text{Pm}^{\text{RU}_2}, .., \text{Pm}^{\text{RU}_5}$. Note that PASCAL VOC 2012 stage-wise results are without using CRF or ResNet-101. Fig. 4.11 shows the stage-wise performance (in terms of mean IoU (%)) of G-FRNet and LRN [3] on the datasets used in this work. Recall that the difference between G-FRNet and LRN are the gate units. From this analysis, it is clear that the inclusion of gate units not only improves the overall performance of the network but also achieves performance gains at each stage of the feedback refinement network.

Stages	CamVid		PASCAL VOC 2012		SUN-RGBD	
	LRN	G-FRNet	LRN	G-FRNet	LRN	G-FRNet
P_m^G	50.9	54.5	58.4	64.1	32.67	31.0
$P_m^{RU_1}$	55.5	61.3	61.6	66.6	33.91	31.6
$P_m^{RU_2}$	59.1	65.2	61.9	68.1	34.54	32.4
$P_m^{RU_3}$	60.9	67.1	62.6	68.3	35.6	32.7
$P_m^{RU_4}$	61.4	67.8	62.5	68.6	36.31	32.8
$P_m^{RU_5}$	**61.7**	**68.0**	**62.8**	**68.7**	**36.86**	**33.1**

(a)

Stages	Horse Parsing		Cow Parsing		Pascal-Person-Part	
	LRN	G-FRNet	LRN	G-FRNet	LRN	G-FRNet
P_m^G	60.6	66.42	56.22	60.35	60.51	58.79
$P_m^{RU_1}$	64.73	68.83	60.15	62.59	62.36	59.2
$P_m^{RU_2}$	64.78	70.05	61.87	64.61	63.6	59.7
$P_m^{RU_3}$	65.78	70.70	62.46	65.15	63.97	60.10
$P_m^{RU_4}$	65.81	70.79	62.49	65.2	64.11	60.3
$P_m^{RU_5}$	**65.89**	**70.83**	**62.53**	**65.35**	**64.61**	**60.75**

(b)

Table 4.7: (a) Stage-wise mean IoU on PASCAL VOC 2012 validation set, CamVid, and SUN-RGBD dataset. (b) Stage-wise mean IoU on Horse-Cow parsing and Pascal-Person-Part dataset. Note that for Pascal-Person-Part stage-wise numbers are reported for G-FRNet-Res101 and LRN-Res101

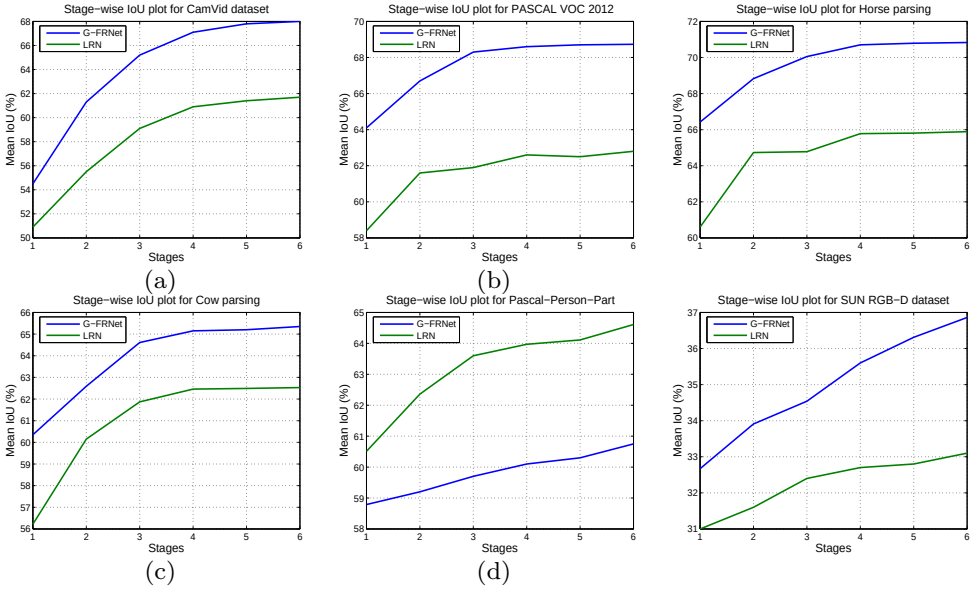


Figure 4.11: Comparison of stage-wise mean IoU on (a) CamVid dataset; (b) PASCAL VOC 2012 validation set (c) Horse parsing (d) Cow parsing (e) Pascal-Person-Part and (f) SUN-RGBD dataset between LRN [3] and proposed network G-FRNet. Note that Pascal-Person-Part stage-wise results are using G-FRNet-Res101 and LRN-Res101.

4.4.8 Exploration Studies

From the qualitative results shown in Fig. 4.7 Fig. 4.8, Fig. 4.9, Table 4.5 (b), and Table 4.6 (b) we can see that our predictions are more precise and semantically

meaningful than the baselines. For example, smaller regions (e.g. tail) in the horse-cow parsing dataset and thinner objects (e.g. column-pole, pedestrian, sign-symbol) in the CamVid dataset can be precisely labeled by G-FRNet. G-FRNet is also capable of effectively handling categories that similar in visual appearance (e.g. horse and cow). Regions with similar appearance (e.g. body parts of horse and cow) can be discriminated by the global contextual guidance provided by the gate units. The local boundaries for different semantic regions are preserved using the low-frequency information from earlier layers.

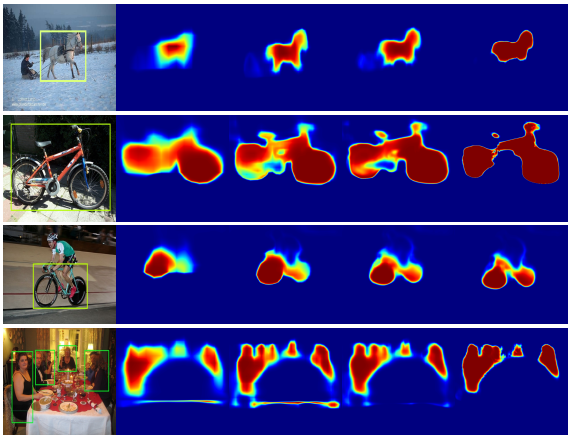


Figure 4.12: Class-wise heatmap visualization on PASCAL VOC 2012 validation set images after each stage of refinement. Interestingly, the network gradually aligns itself more precisely with semantic labels, while correcting initially mislabeled regions. The rightmost column shows the heatmap of the final prediction layer.

Fig. 4.12 shows that prediction quality progressively improves with each successive stage of refinement. In coarse-level predictions, the network is only able to identify some parts of objects or semantic categories. With each stage of gated refinement, missing parts of the object are recovered and mislabeled parts are corrected.

Fig. 4.13 shows a comparison between different methods in terms of the total number of model parameters and mean IoU (%) on PASCAL VOC 2012 dataset.

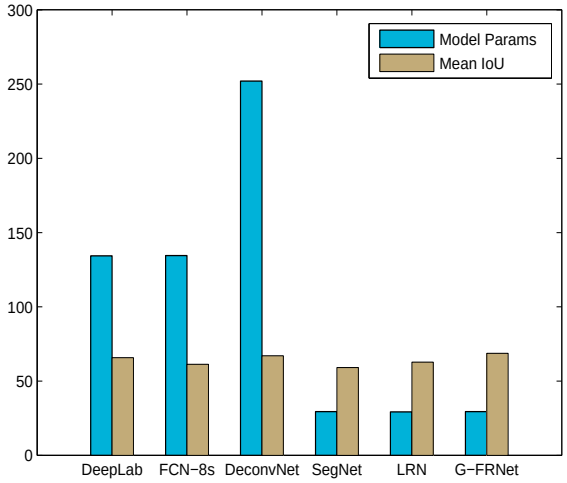


Figure 4.13: Analysis on the number of model parameters (in millions) and the mean IoU (%) on PASCAL VOC 2012 validation set for different methods. The rightmost method is our proposed model, which achieves best performance, even with considerably fewer parameters, and a more parsimonious model structure.

Although our model has only 12 to 25 percent of the number of parameters of other state-of-the-art methods (FCN [5] and DeconvNet [8]), it achieves very competitive performance. This shows the efficiency of the proposed model despite its simplicity and also the broader value of the proposed gating mechanism.

Fig. 4.14 shows dense predictions on challenging images of CamVid dataset. The labeling of finer details such as the column pole are improved. This improvement is mainly due to the recurrent connection of the weights in the encoder and decoder networks respectively. In addition, top-down modulation through gate units significantly helps to select relevant low-level features towards recovery of fine spatial details (especially for smaller objects) even after lowering the resolution through several pooling layers.

Additionally, the value of the gating mechanism is demonstrated in each of the experiments, with its strengths evident in both the qualitative and quantitative results.

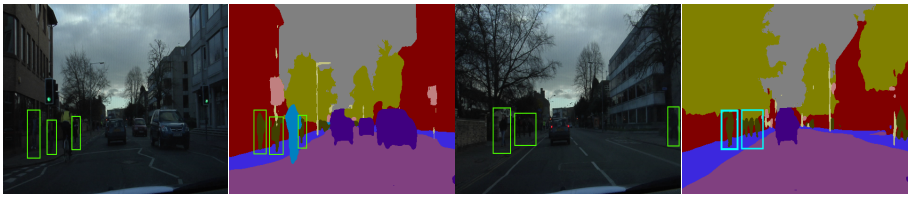


Figure 4.14: Qualitative analysis of dense predictions on a few challenging object categories in CamVid dataset. For example, pedestrians are labeled accurately by our model, despite the varying light settings.

The LRN method uses the upper layer feature map alone. We reported the result of LRN for all datasets. It is clear that the proposed gating mechanism in G-FRNet significantly improves performance compared with LRN. As higher layers see a larger part of the scene and represent more complex concepts (while also being composed of features from earlier layers) it is natural that subsequent layers are able to resolve ambiguity that preceding layers cannot possibly resolve. Our work is the first that uses a gating mechanism in the encoder-decoder network for dense image labeling. Given the apparent efficacy of the proposed gating mechanism, we expect that this work will inspire significant interest in exploring different gating mechanisms within future work. It is especially noteworthy how powerful this architectural modification is shown to be across a wide array of different datasets, with different properties and labels. With respect to the mechanics of the gating mechanism, we have also tried a variety of alternative design choices. When we use additive interaction in gate units, the result is **66.76%** mean IoU on the PASCAL validation set. In comparison, our proposed method with an element-wise product yields **68.7%** mean IoU on PASCAL val set. Intuitively, a multiplicative mechanism allows for strong modulation of representations deemed to be incorrect by higher layer features. In the additive case, while activation may be boosted for the correct representations by higher layer features,

there remains a residue of incorrect representations that may weaken final predictions. Multiplicative gating is demonstrably valid from a performance standpoint, but also intuitive in that it provides a stronger capacity to resolve ambiguity present among earlier layers.

Chapter 5

Salient Object Detection

In this chapter we propose a new approach for salient object detection. Recent success of encoder-decoder networks in pixel-wise labeling tasks, we apply a network with this structure to detect salient regions in an end-to-end fashion. This takes the form of our proposed context-aware refinement network wherein the decoder part takes coarse saliency maps generated by the encoder network and hierarchically refines the saliency map to produce a final output that matches the resolution of the input. To overcome the limitations with existing approaches, we propose a refinement unit that takes full advantage of the high spatial dimension features from earlier layers in the refinement process. As demonstrated in Fig. 5.1, we observe that high-level features can better locate the salient object and low-level features capture rich spatial information. With that being said, we believe that integrating high-level features with low-level features is useful in the salient object detection task. In the following sections we describe the different components of pur proposed network.

Figure 5.1: An example of applying context-aware refinement network to an initial saliency map produced by the encoder network. Compared to initial saliency map, the refined saliency map has significantly sharper edges and also has better spatial information.

5.1 Context-Aware Refinement Network

In this section, we discuss our proposed context-aware refinement network to address the problem of salient object detection. Then, we design a fully-convolutional feedback refinement network using context-aware refinement units.

5.1.1 Overview

We adopt the popular encoder-decoder network architecture for salient object detection, where a CNN initially encodes the input image to produce a coarse spatial resolution prediction, and then a refinement network decodes the coarse saliency map to provide a full resolution pixel-wise prediction map. Our overall salient object detection network is illustrated in Fig. 5.2. We employ pre-trained ResNet-101 [19] as the encoder network, and we propose a novel refinement network that serves as our decoder network. We extract multi-scale feature maps from different stages of the encoder. The context-aware refinement network uses these feature maps to generate semantic score maps in each stage of the refinement network. Similar to previous approaches [3; 36; 7], semantic score maps for lower resolutions are upsampled through bilinear interpolation and combined with the feature maps from the encoder

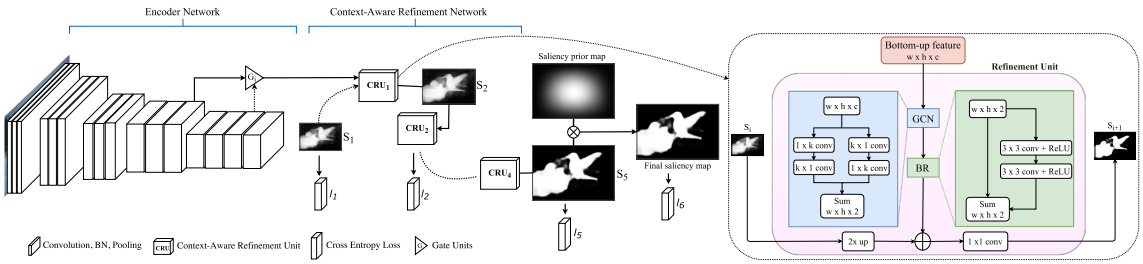


Figure 5.2: An overview of the proposed salient object detection framework. The bottom-up (encoder) network consisting of multiple layers (e.g. convolution, batch normalization, pooling, ReLU) is integrated with the refinement network through skip connections. The refinement network has context-aware refinement units (CRU_1, CRU_2,..., CRU_4) that take a bottom-up feature map and previous stage prediction map (S_i) to generate subsequent stage prediction maps (S_{i+1}). We down-sampled the ground-truth saliency maps to incorporate stage-wise supervision (l_1, l_2,..., l_6) in the refinement network. GCN and BR are used in CRUs (see main text for details). Note that we also combine the final prediction map with the saliency prior to obtain the final saliency map.

to generate a higher resolution refined saliency map. In our case, this combination is influenced by the refinement units involved. The semantic score map generated from the last stage of the refinement unit is treated as the final prediction map of our network. In addition, the saliency prior map is integrated with the final prediction map as a final stage of refinement before evaluating the loss function. In the following section, we discuss the context-aware refinement unit and saliency prior map.

5.1.2 Context-Aware Refinement Unit

The task of salient object detection requires per-pixel classification as well as correct localization. Current state-of-the-art salient object detection methods [7; 79] mostly target the design principles of the decoding process such that an initial prediction map is refined to produce the full resolution map. However, in most cases, refinement is done across different levels by combining convolution features from the

encoder (or feature extractor) network with decoder layers. Directly combining convolution features with the semantic score map through concatenation or element-wise summation may have unpredictable consequences, and has the possibility of degrading the contribution of lower depth feature maps (semantic score maps). Hence, in only using features from the encoder network through skip connections [5; 3] there are inherent limits on spatial detail that may be recovered since the model cannot take full advantage of the higher resolution feature maps. Therefore, following previous work [80], we integrate a multiplicative gate unit at each stage of refinement that controls the information being passed forward to resolve ambiguity between background and salient object classes.

Moreover, in salient object detection, the object is often biased in its position towards the center of the image and the classifier has a view of the entire object in context only within the deepest layers of the encoder. However, if the salient object is resized to a large scale, then the receptive field (kernel) of the skip connections covers only a part of the object, which can be problematic in refining missing details. In [80], all the skip connections in gate units and refinement units use a 3×3 receptive field to generate semantic score maps.

Based on the above observations, and also drawing inspiration from [81], we design a refinement unit that is mainly composed of a Global Convolution Network (GCN) and a Boundary Refinement (BR) block that overcomes these drawbacks. GCN uses a combination of $1 \times k + k \times 1$ and $k \times 1 + 1 \times k$ convolutions, resulting in a $k \times k$ convolution that enables dense connection within the $k \times k$ region (instead of directly using a $k \times k$ kernel), and thereby helping to capture broader context. BR consists of

stack of two 3×3 convolutions followed by element-wise summation to further refine the boundary pixels. We now describe the detailed architecture of context-aware refinement units (CRU).

The detailed architecture of the CRU is illustrated in the dotted box of Fig. 5.2 which has two input paths. Our refinement units are generic and can be modified to accept an arbitrary number of feature maps with different resolutions. Note that, although these units are identical, they do not share weights among them since each unit learns to recover missing spatial information in order to resolve ambiguity during refinement stages. It is also noteworthy that each CRU combines feature representations obtained from different levels of the encoder network.

The first input of each refinement unit is comprised of a bottom-up feature map derived from a multiplicative gate unit that serves a primary role of filtering out ambiguity between background and salient objects by controlling the activation from features passed from encoder layers to decoder layers. The saliency map predicted from the prior stage S_i serves as the second input to the refinement unit. To that end, the first input is passed sequentially through a global convolution unit and boundary refinement unit before being combined with the 2x upsampled saliency map derived from the prior stage through concatenation followed by a 1×1 convolution across layers. The formulation of getting a bottom-up feature map from a gate unit is described by the following equations:

$$v_i = T_f(C_g^{i+1}), \;\; u_i = T_f(C_g^i), \;\; Z_f^i = v_i \odot u_i \tag{5.1}$$

where $\odot$ denotes an element-wise product. Note that, C_g^i and C_g^{i+1} are the feature map from subsequent layers in the encoder which are passed through a transformation

function T_f to map these to semantic score maps. As noted earlier, Z_f^i is then fed to the refinement unit as the first input.

In summary, the refinement unit at each stage takes the bottom-up feature Z_f^i and last stage prediction map S_i as inputs and generates the next stage prediction map S_{i+1} through the series of operations mentioned earlier. The operations inside each refinement unit are as follows:

$$S_{i+1} = \mathbb{C}_{1\times1}(\varrho(\phi(Z_f^i)) \oplus \mathbb{U}(S_i)) \tag{5.2}$$

where $\mathbb{C}$, ϱ, ϕ, $\oplus$, and $\mathbb{U}$ denotes 1×1 convolution, global convolution unit, boundary refinement unit, concatenation, and 2x upsample operation respectively.

5.1.3 Saliency Prior Map

We also integrate a saliency prior map as an additional input to the network. We first calculate the per-pixel average of ground-truth training images which serves as the saliency prior map for the network. If pixels marked salient were uniformly distributed, this prior would have no effect. However, salient objects tend to be near the center of the image (likely due to compositional bias) in a manner determined by the purpose of dataset and how it was composed. Taking this into consideration, it is important to model such spatial bias and we do so by creating a prior distribution S_p that is multiplied element-wise with the final predicted saliency map. We convolve the final prediction layer with a Gaussian to regularize the predictions. Since the final prediction layer has two output channels (foreground and background), we slice the feature map to separate them. Note that only foreground feature slices are combined with the prior map since these contain the objectness score for each pixel that

corresponds roughly to different semantic categories. We summarize the operations as follows:

$$S_p(i,j) = \frac{1}{N}\sum_{m=1}^{N}\sum_{i=1}^{h}\sum_{j=1}^{w} S_m(i,j) \qquad S_i' = S_i \times G_\sigma \qquad S_i'' = S' \odot S_p \tag{5.3}$$

5.1.4 Training with Multi-stage Supervision

Inspired by [3; 7; 32], we apply multi-stage supervision in our end-to-end network. More specifically, assume $I_m \in \mathbb{R}^{h\times w\times d}$ to be a training instance with ground-truth saliency mask $S_m \in \mathbb{R}^{h\times w}$. We obtain m resized ground-truth maps $(R_1, R_2, ..., R_m)$ by resizing S_m. A loss function ϕ_i (pixel-wise cross entropy loss) is defined to measure the quality of predicted saliency map against the resized ground-truth saliency mask $R_i(S_m)$ at different stages of the refinement network. We can write these operations as follows:

$$\ell = \sum_{m=1}^{5} l_m \qquad l_m = \xi\Big(R_i, S_{m_i}\Big) \qquad \xi = \frac{1}{N}\sum_{i} p\log(x_i, y_i|I_i) \tag{5.4}$$

where ξ denotes cross-entropy loss at each stage. The final loss ℓ is the summation of cross-entropy losses across different stages of the refinement network. We train the network end-to-end using back-propagation to optimize the final loss.

5.2 Experiments and Results

To demonstrate the effectiveness of each component of our network architecture, and study the performance of our proposed approach, we present results from experiments on four salient object detection benchmark datasets and show quantitative and qualitative comparisons of our methods with recent state-of-the-art methods. In

the following section, we firstly describe the implementation specific details. Then, we report performance on several saliency detection benchmarks followed by analysis of different variants of our approach.

5.2.1 Implementation Details

Our network is implemented based on the publicly available Caffe library [54]. We use a single GTX Titan X GPU for both training and testing. Inspired by [24], we use the "poly" learning rate policy. Taking training efficiency into consideration, the mini-batch size is set to 1, and loss is updated after every 10 iterations (i.e. each image is used 10 times for training). We train the network using stochastic gradient descent with momentum of 0.9, and weight decay of 0.0005. The total number of iterations is set to 20k. The weights of all the newly added convolution layers in the refinement network are randomly initialized from a standard normal distribution. Since we use the pre-trained ResNet-101 model for initializing the encoder part of our network, we normalize the data using the mean and standard deviation from VGG-16. We use pixel-wise cross entropy loss to optimize the network. As commonly done in the training procedure (due to hardware constraints), we perform random cropping of the images. During training, crop size is set to 321×321. Since all the proposed modules in our network can handle input images of different sizes, we test our network with the full resolution image. To show the effectiveness of our method and the merit of individual components, we carry out comprehensive experiments including ablation studies. We report performance for the following variants of our model including the baseline:

G-FRNet: Gated Feedback Refinement Network [80] that includes the gating mechanism prior to passing information to the refinement units. We consider G-FRNet as our base model and report its experimental results.

CARNet: Context-Aware Feedback Refinement Network built on top of G-FRNet [80] for salient object detection. We integrate the prior map within the training procedure.

CARNet†: This is the same as CARNet except that we add the global convolution network (GCN) and boundary refinement (BR) block within the refinement process.

5.2.2 Datasets and Evaluation Metrics

Datasets: We follow the training protocol suggested in [7]. More specifically, we use the MSRA-10K [82] dataset for training and evaluating our proposed method on four saliency detection benchmark datasets, including PASCAL-S [83], ECSSD [84], HKU-IS [49], and DUT-OMRON [85]. MSRA-10K dataset consists of 10,000 images with pixel-wise annotation for salient objects. PASCAL-S dataset contains 850 images with multiple complex objects derived from PASCAL VOC 2012 validation set that provides saliency estimates in the [0, 1] range. As suggested by the author of this dataset, we threshold the saliency values using a threshold of 0.5 to obtain the binary object mask. HKU-IS dataset provides 4,447 complex images with low-contrast and multiple salient objects in each image. Similarly, DUT-OMRON is a large dataset which contains 5,168 challenging images (one or more salient objects) with complex backgrounds.

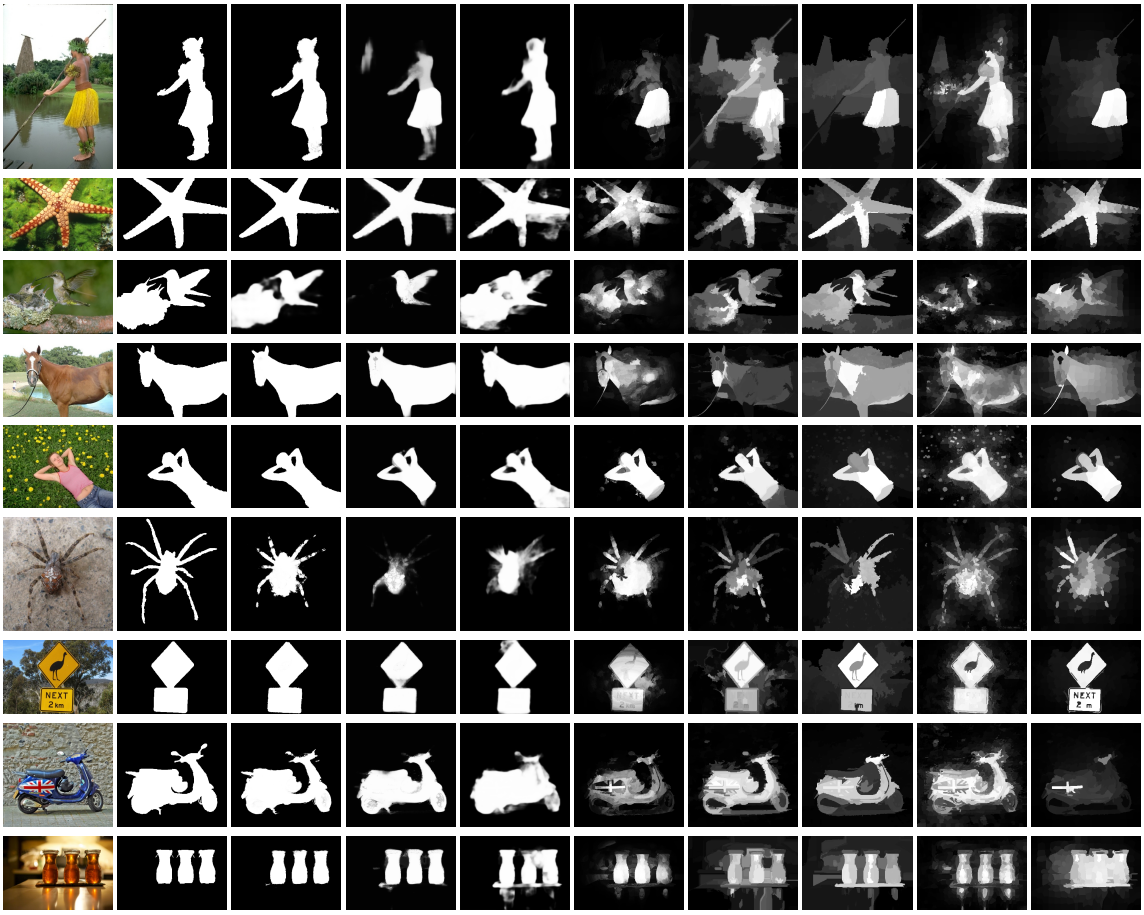
*	ECSSD [84]		HKU-IS [86]		PASCAL-S [83]		DUT-OMRON [85]	
	F-measure	MAE	F-measure	MAE	F-measure	MAE	F-measure	MAE
RC [47]	0.741	0.187	0.726	0.165	0.640	0.225	-	-
DSR [87]	0.737	0.173	0.735	0.140	0.646	0.204	-	-
DRFI [46]	0.787	0.166	0.783	0.143	0.679	0.221	0.664	0.150
MDF [49]	0.833	0.108	0.860	0.129	0.764	0.145	0.640	0.092
CHM [88]	0.722	0.195	0.728	0.158	0.631	0.222	-	-
MC [51]	0.822	0.107	0.781	0.098	0.721	0.147	0.703	0.088
ELD [89]	0.865	0.981	0.844	0.071	0.767	0.121	0.719	0.091
RFCN [53]	0.898	0.097	0.895	0.079	0.827	0.118	0.747	0.095
DHSNet [7]	0.905	0.061	0.892	0.052	0.820	0.091	0.740	-
DCL [52]	0.898	0.071	0.907	0.048	0.822	0.108	0.757	0.080
DSS [79]	0.915	0.052	0.913	0.039	0.830	0.080	-	-
CARNet†	0.9250	0.040	0.912	0.034	0.8341	0.086	0.7895	0.061

Table 5.1: Quantitative comparison (in terms of average F_β and MAE) with state-of-the-the methods. Best and second best scores are shown in red and blue text respectively.

Evaluation Metrics: Following previous work [7], we use four different standard metrics to measure the performance including precision-recall (PR) curves, F-measure, mean absolute error (MAE), and area under ROC curve (AUC). We calculate the precision and recall curve by thresholding the predicted saliency map using a set of thresholds, and compare the predicted binary map with the ground-truth map. MAE is the average pixel-wise difference between the predicted saliency map and the binary ground-truth map. We set β^2 in F-measure to 0.3 following previous works.

5.2.3 Performance Comparison with State-of-the-art Methods

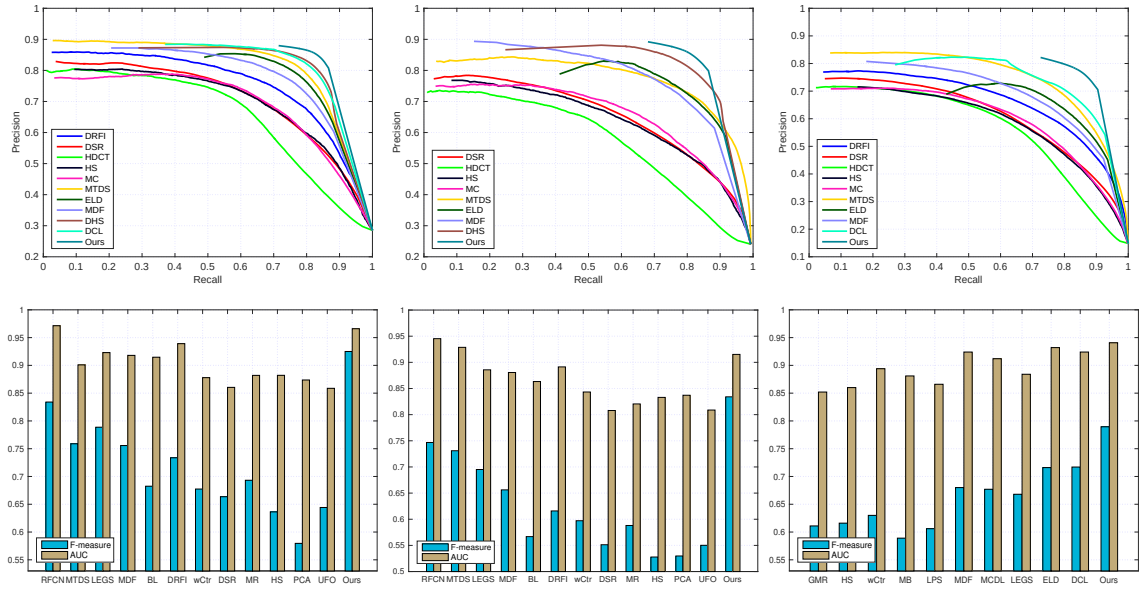
We compare our proposed salient object detection model with state-of-the-art methods, including DSS [79], RFCN [53], DCL [52], DHS [7], MTDS [90], DRFI [46], LEGS [50], MDF [49]. The first few approaches are recent deep learning methods.



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Figure 5.3: Visual comparison of saliency maps with state-of-the-art methods, including our approach. (a) Input image (b) ground truth (c) CARNet (d) DCL (e) DHS (f) DSR (g) DRFI (h) HS (i) HDCT (j) MC. Our approach consistently produces saliency maps closest to the ground truth.

Initially, we compare our approach with existing methods in terms of F-measure and MAE scores as shown in Table 5.1. Our approach achieves the best performance for most of the datasets. Our proposed approach is capable of not only detecting salient objects of different scales but also generating precise saliency maps in challenging scenarios (see Fig. 5.3). Fig. 5.4 presents the comparison of our method with state-of-the-art methods through PR-curves, F-measure and AUC metrics. It is clear from Fig. 5.4 and Table 5.1 that our proposed approach outperforms the existing methods



(a) ECSSD (b) PASCAL-S (c) DUT-OMRON

Figure 5.4: Comparison with state-of-the-art salient object detection methods on 3 different datasets. For each dataset, the first row shows the precision-recall curves and second row shows the F-measure and AUC. Our proposed approach CARNet† consistently outperforms other methods across all the datasets. In particular, the PR-curves show that our approach achieves significantly higher precision with higher recall, which demonstrates that our method locates salient objects more accurately and precisely. Note that DHSNet [7] includes the test set of DUT-OMRON in its training data. Therefore, we do not include it in the comparison based on the DUT-OMRON dataset.

with a reasonable margin.

5.2.4 Comparison with Different Variants

To demonstrate in greater detail the role of different components of our proposed network, we report performance of different variants (Sec. 5.2.1) of our network in Table 5.2. GFRNet is our base model, whereas CARNet is G-FRNet with spatial prior information. CARNet performs better than GFRNet due to the fact that adding

prior to the network refines expectation based on interaction between spatial position and features, and thus helps providing a more precise final prediction. To further improve the performance of CARNet, we integrate GCN and BR within CARNet (i.e. CARNet†). Our final model CARNet† achieves the best performance and this gain in performance can be attributed to the improvement in labeling capability induced by GCN and BR.

*	HKU-IS [86]		ECSSD [84]		PASCAL-S [83]		DUT-OMRON [85]	
	F-measure	AUC	F-measure	AUC	F-measure	AUC	F-measure	AUC
G-FRNet [80]	0.9085	0.9635	0.9080	0.9560	0.8310	0.9116	0.7840	0.9363
CARNet	0.9109	0.9657	0.9095	0.9567	0.8320	0.9142	0.7870	0.9405
CARNet†	**0.9115**	**0.9660**	**0.9250**	**0.9598**	**0.8341**	**0.9152**	**0.7895**	**0.9407**

Table 5.2: Comparison of different variants of our proposed approach. Our final model CARNet† achieves the best performance when compared to other variants of our model.

Chapter 6

Conclusion and Future Work

In this thesis, we have presented novel deep learning based methods for dense image labeling tasks. We have made three significant contributions to research in the area of dense image labeling. First, we proposed an approach based on pixel-wise class label assignments at a coarse level of abstraction that can produce semantically accurate predictions. The novelty of this approach is the integration of deep convolutional neural networks with image-specific weighted support vector classification, and demonstration of the value in leveraging distinct and heterogeneous datasets. Second, we have presented a novel end-to-end deep learning framework for dense image labeling deemed a coarse-to-fine gated feedback refinement network. This model produces segmentation labels in a coarse-to-fine manner. The segmentation labels at coarse levels are used to progressively refine the labeling produced at finer levels. Third, we have introduced a novel end-to-end refinement based architecture combined with prior information for solving the problem of salient object detection. The most important contribution of this work is the stage-wise saliency map refinement, which

results in precise saliency map. In addition, experimental results based on ablation analysis reveal generality in the value of coarse-to-fine predictions, deep supervision, skip connections, and gated refinement with the implication that a wide array of canonical neural network architectures may benefit from these simple architectural modifications.

Many interesting research questions arise from the approach and results presented in this thesis. One especially fruitful avenue for further investigation is to remove focus on the correctness of predictions possible at intermediate stages given that skip connections and gating allow for repair or modulation of erroneous representations. In practice, this suggests the possibility of error correcting iterative gated refinement as an interesting and important direction for future work, which may also allow for networks with a stronger representational capacity made possible by the additional slack afforded by ambiguity resolving mechanisms proposed.

Bibliography

[1] S. Gould, R. Fulton, and D. Koller, "Decomposing a scene into geometric and semantically consistent regions," in *CVPR*, 2009. v, ix, 12, 19, 20, 21, 22

[2] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010. vi, ix, 12, 19, 23, 24, 43, 47

[3] M. A. Islam, S. Naha, M. Rochan, N. Bruce, and Y. Wang, "Label refinement network for coarse-to-fine semantic segmentation," *arXiv:1703.00551v1*, 2017. vi, vii, 37, 39, 42, 43, 44, 45, 46, 47, 49, 50, 51, 52, 58, 60, 63

[4] A. Kundu, V. Vineet, and V. Koltun, "Feature space optimization for semantic video segmentation," in *CVPR*, 2016. vii, 42, 43, 44

[5] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015. vii, 2, 6, 7, 12, 23, 30, 31, 32, 34, 36, 42, 43, 44, 45, 46, 50, 54, 60

[6] J. Wang and A. L. Yuille, "Semantic part segmentation using compositional model combining shape and appearance," in *CVPR*, 2015. vii, ix, 47, 48, 49

[7] N. Liu and J. Han, "Dhsnet: Deep hierarchical saliency network for salient object detection," in *CVPR*, 2016. viii, 6, 10, 58, 59, 63, 65, 66, 68

[8] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *ICCV*, 2015. ix, 2, 6, 8, 23, 29, 30, 31, 32, 42, 44, 45, 46, 50, 54

[9] Y. Wang, J. Liu, Y. Li, J. Yan, and H. Lu, "Objectness-aware semantic segmentation," in *ACMMM*, 2016. ix, 44

[10] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *CVPR*, 2015. x, 49, 50

[11] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," in *ICLR*, 2015. 2, 6, 7, 14, 18, 23, 42, 43, 44, 45, 46, 47, 49, 50

[12] K. Grill-Spector and N. Kanwisher, "Visual recognition as soon as you know it is there, you know what it is," *Psychological Science*, vol. 16, no. 2, pp. 152–160, 2005. 2

[13] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for scene segmentation," *TPAMI*, 2017. 2, 6, 8, 29, 30, 31, 32, 41, 42, 43, 45, 46, 50

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012. 6

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015. 6

[16] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatialtemporal features with 3d convolutional networks," in *ICCV*, 2015. 6

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015. 6, 7, 14, 31, 40

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015. 6, 8

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 6, 58

[20] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scence labeling," *TPAMI*, 2013. 6

[21] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *CVPR*, 2015. 6, 8, 30, 34, 45, 47

[22] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich, "Feedforward semantic segmentation with zoom-out features," in *CVPR*, 2015. 6, 20, 21, 23, 45

[23] M. A. Islam, N. Bruce, and Y. Wang, "Dense image labeling using deep convolutional neural networks," in *CRV*, 2016. 6

[24] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *arXiv:1606.00915*, 2016. 7, 41, 44, 45, 46, 48, 49, 64

[25] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr, "Conditional random fields as recurrent neural networks," in *ICCV*, 2015. 7, 23, 44, 45

[26] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR*, 2016. 7, 42, 43, 45

[27] P. J. Burt and E. H. Adelson, "The laplacian pyramid as a compact image code," *TC*, 1983. 7

[28] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *ICCV*, 2015. 7, 41

[29] E. Denton, S. Chintala, A. Szlam, and R. Fergus, "Deep generative image models using a laplacian pyramid of adversarial networks," in *NIPS*, 2015. 8

[30] S. Honari, J. Yosinski, P. Vincent, and C. Pal, "Recombinator networks: Learning coarse-to-fine feature aggregation," in *CVPR*, 2016. 8

[31] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan, "Pixelnet: Towards a general pixel-level architecture," *arXiv:1609.06694*, 2016. 8

[32] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *AISTATS*, 2015. 8, 39, 63

[33] S. Xie and Z. Tu, "Holistically-nested edge detection," in *ICCV*, 2015. 8

[34] N. Liu and J. Han, "Dhsnet: Deep hierarchical saliency network for salient object detection," in *CVPR*, 2016. 8

[35] G. Ghiasi and C. C. Fowlkes, "Laplacian pyramid reconstruction and refinement for semantic segmentation," in *ECCV*, 2016. 8, 45

[36] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár, "Learning to refine object segments," in *ECCV*, 2016. 8, 36, 58

[37] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," *arXiv:1611.06612*, 2016. 8, 48

[38] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," *arXiv:1611.00850*, 2016. 8

[39] M. M. Chun and Y. Jiang, "Top-down attentional guidance based on implicit learning of visual covariation," *Psychological Science*, vol. 10, no. 4, pp. 360–365, 1999. 9

[40] A. Gazzaley and A. C. Nobre, "Top-down modulation: bridging selective attention and working memory," *Trends in cognitive sciences*, vol. 16, no. 2, pp. 129–135, 2012. 9

[41] J. B. Hopfinger, M. H. Buonocore, and G. R. Mangun, "The neural mechanisms of top-down attentional control," *Nature neuroscience*, vol. 3, no. 3, pp. 284–291, 2000. 9

[42] J. T. Serences and S. Yantis, "Selective visual attention and perceptual coherence," *Trends in cognitive sciences*, vol. 10, no. 1, pp. 38–45, 2006. 9, 28

[43] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *TPAMI*, vol. 20, no. 11, pp. 1254–1259, 1998. 9

[44] Y. Xie, H. Lu, and M.-H. Yang, "Bayesian saliency via low and mid level cues," *TIP*, vol. 22, no. 5, pp. 1689–1698, 2013. 9

[45] D. A. Klein and S. Frintrop, "Center-surround divergence of feature statistics for salient object detection," in *ICCV*, 2011. 9

[46] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li, "Salient object detection: A discriminative regional feature integration approach," in *CVPR*, 2013. 9, 66

[47] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu, "Global contrast based salient region detection," *TPAMI*, vol. 37, no. 3, 2015. 9, 66

[48] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum, "Learning to detect a salient object," *TPAMI*, vol. 33, no. 2, pp. 353–367, 2011. 9

[49] G. Li and Y. Yu, "Visual saliency based on multiscale deep features," in *CVPR*, 2015. 65, 66

[50] L. Wang, H. Lu, X. Ruan, and M.-H. Yang, "Deep networks for saliency detection via local estimation and global search," in *CVPR*, 2015. 10, 66

[51] R. Zhao, W. Ouyang, H. Li, and X. Wang, "Saliency detection by multi-context deep learning," in *CVPR*, 2015. 6, 10, 66

[52] G. Li and Y. Yu, "Deep contrast learning for salient object detection," in *CVPR*, 2016. 6, 10, 66

[53] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan, "Saliency detection with recurrent fully convolutional networks," in *ECCV*, 2016. 6, 10, 66

[54] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv:1408.5093*, 2014. 14, 18, 40, 64

[55] X. Yang, Q. Song, and Y. Wang, "A weighted support vector machine for data classification," *IJPAI*, vol. 21, no. 05, pp. 961–976, 2007. 17

[56] F. Liu, G. Lin, and C. Shen, "Crf learning with cnn features for image segmentation," *Pattern Recognition*, vol. 48, no. 10, pp. 2983–2992, 2015. 20, 21

[57] V. Lempitsky, A. Vedaldi, and A. Zisserman, "Pylon model for semantic segmentation," in *NIPS*, 2011. 21

[58] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *PAMI*, vol. 35, no. 8, pp. 1915–1929, 2013. 21

[59] A. Sharma, O. Tuzel, and D. W. Jacobs, "Deep hierarchical parsing for semantic segmentation," in *CVPR*, 2015. 21

[60] R. Mohan, "Deep deconvolutional networks for scene parsing," *arXiv:1411.4101*, 2014. 21

[61] B. Wonmin, T. Breuel, F. Raue, and M. Liwicki, "Scene labeling with lstm recurrent neural networks," in *CVPR*, 2015. 21

[62] G. E. Hinton, J. L. McClelland, and D. E. Rumelhart, "Distributed representations," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 1986. 26

[63] C. W. Eurich and H. Schwegler, "Coarse coding: calculation of the resolution achieved by a population of large receptive field neurons," *Biological cybernetics*, vol. 76, no. 5, pp. 357–363, 1997. 26

[64] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground-truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009. ix, 42

[65] F. Visin, M. Ciccone, A. Romero, K. Kastner, K. Cho, Y. Bengio, M. Matteucci, and A. Courville, "Reseg: A recurrent neural network-based model for semantic segmentation," in *CVPR Workshops*, 2016. 42, 43

[66] P. Sturgess, K. Alahari, L. Ladicky, and P. H. Torr, "Combining appearance and structure from motion features for road scene understanding," in *BMVC*, 2009. 42

[67] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," *arXiv:1611.09326*, 2016. 42, 43

[68] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, "Semantic image segmentation via deep parsing network," in *ICCV*, 2015. 44, 45

[69] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, "Attention to scale: Scale-aware semantic image segmentation," in *CVPR*, 2016. 44, 45, 46, 49

[70] R. Zhang, W. Yang, Z. Peng, X. Wang, and L. Lin, "Progressively diffused networks for semantic image segmentation," *arXiv:1702.05839*, 2017. 44

[71] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *ICCV*, 2011. 43

[72] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *NIPS*, 2011. 46

[73] S. Tsogkas, I. Kokkinos, G. Papandreou, and A. Vedaldi, "Deep learning for semantic part segmentation with high-level guidance," *arXiv:1505.02438*, 2015. 47

[74] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. L. Yuille, "Joint object and part segmentation using deep learned potentials," in *ICCV*, 2015. 47

[75] X. Liang, X. Shen, D. Xiang, J. Feng, L. Lin, and S. Yan, "Semantic object parsing with local-global long short-term memory," in *CVPR*, 2016. 47, 49

[76] F. Xia, P. Wang, L.-C. Chen, and A. L. Yuille, "Zoom better to see clearer: Human and object parsing with hierarchical auto-zoom net," in *ECCV*, 2016. 49

[77] X. Liang, X. Shen, J. Feng, L. Lin, and S. Yan, "Semantic object parsing with graph lstm," in *ECCV*, 2016. 49

[78] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," *arXiv:1511.02680*, 2015. 42, 50

[79] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. Torr, "Deeply supervised salient object detection with short connections," in *CVPR*, 2017. 6, 59, 66

[80] M. A. Islam, M. Rochan, N. Bruce, and Y. Wang, "Gated feedback refinement network for dense image labeling," in *CVPR*, 2017. 60, 65, 69

[81] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, "Large kernel matters–improve semantic segmentation by global convolutional network," *arXiv preprintarXiv:1703.02719*, 2017. 60

[82] K. Shi, K. Wang, J. Lu, and L. Lin, "Pisa: Pixelwise image saliency by aggregating complementary appearance contrast measures with spatial priors," in *CVPR*, 2013. 65

[83] Y. Li, X. Hou, C. Koch, J. M. Rehg, and A. L. Yuille, "The secrets of salient object segmentation," in *CVPR*, 2014. 65, 66, 69

[84] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection," in *CVPR*, 2013. 65, 66, 69

[85] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang, "Saliency detection via graph-based manifold ranking," in *CVPR*, 2013. 65, 66, 69

[86] V. Movahedi and J. H. Elder, "Design and perceptual validation of performance measures for salient object segmentation," in *CVPRW*, 2010. 66, 69

[87] X. Li, H. Lu, L. Zhang, X. Ruan, and M.-H. Yang, "Saliency detection via dense and sparse reconstruction," in *ICCV*, 2013. 66

[88] X. Li, Y. Li, C. Shen, A. Dick, and A. Van Den Hengel, "Contextual hypergraph modeling for salient object detection," in *ICCV*, 2013. 66

[89] G. Lee, Y.-W. Tai, and J. Kim, "Deep saliency with encoded low level distance map and high level features," in *CVPR*, 2016. 6, 66

[90] X. Li, L. Zhao, L. Wei, M.-H. Yang, F. Wu, Y. Zhuang, H. Ling, and J. Wang, "Deepsaliency: Multi-task deep neural network model for salient object detection," *TIP*, vol. 25, no. 8, pp. 3919–3930, 2016. 66