Scene segmentation with deep neural networks

Shuai, Bing

2018

Shuai, B. (2018). Scene segmentation with deep neural networks. Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/73331>

<https://doi.org/10.32657/10356/73331>

Scene Segmentation with Deep Neural Networks

Bing Shuai

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2017

*To my parents, my wife, my sister
who always give me love and support.*

Acknowledgments

First and most, thank you to my two advisors Dr. Gang Wang and Professor Kim Hui Yap. They have constantly supported and encouraged me throughout my Ph.D life in Nanyang Technological University. Especially as my research mentor, Dr. Gang Wang opened the door for me to enter the fantastic world of deep learning. This thesis would have not been possible without their constant encouragement and support.

Thank you to School of Electrical and Electronic Engineering, NTU for awarding me the research scholarship and providing me with excellent environment and facilities for the research.

Thank you to my thesis advisory committee (TAC) members Professor Xudong Jiang and Professor Jianfei Cai for their great support and encouragement.

Thank you to my awesome research collaborators: Dr. Zhen Zuo, Dr. Bing Wang, Dr. Ting Liu, Dr. Rahul Rama Varior, Mrs. Abrar Abdul Nabi, Dr. Lifan Zhao, Mr. Henghui Ding, Mr. Jiuxiang Gu, Mr. Ping Hu. They always inspire me to doing in-depth research.

Thank you to my colleagues in Rose lab: Dr. Yan Wang, Dr. Amir Shahroudy, Mr. Jason Kuen, Dr. Jianfang Hu, Mr. Chengde Wan, Mr. Xingxing Wang, Dr. Xiangfei Kong, Dr. Jinghua Wang, Dr. Zhenhua Wang, Dr. Lianyang Ma, Mr. Hongkun Ji, Mr. Jun Liu, Mr. Jianhan Mei, Mr. Zenghai Chen, Dr. Jianlou Si, Mr. Feixiang He, Mr. Jianda Chen and Ms. Kamila Abdiyeva. The Ph.D life would have been much tougher without talking with them ranging from research to personal life.

Finally, I would like to dedicate this thesis to my parents, my wife, my sister and my in-laws who have generously offered me love, support and encouragements.

Contents

Acknowledgments	ii
Abstract	vii
List of Figures	x
List of Tables	xv
List of Abbreviations	xviii
List of Notation	xix
1 Introduction	1
1.1 Challenges and Motivations	2
1.2 Proposed Segmentation Models	5
1.3 Contributions	7
1.4 Thesis Outline	9
2 Literature Review	10
2.1 Object segmentation vs Scene segmentation	10
2.1.1 Object Segmentation	10
2.1.2 Scene Segmentation	12
2.2 Related works	16
2.2.1 Feature Learning	16
2.2.2 Probabilistic Graphical Models	19
2.2.3 Label Transfer Models	21
2.2.4 Model Ensemble	22

2.2.5	Contextual Modeling	23
2.2.6	Recurrent Neural Networks	25
2.2.7	Retaining / Recovering Detailed Information	26
2.2.8	Multi-scale Aggregation	28
2.3	Summary	29
3	Patch-based convolutional networks	31
3.1	Introduction	31
3.1.1	Ensemble Models	34
3.2	Formulation	35
3.3	Local Beliefs	38
3.3.1	Parametric CNNs	38
3.3.2	Data Sampling	39
3.3.3	CNN-Ensemble	41
3.4	Global Beliefs	43
3.4.1	Non-parametric Global Belief Transfer	43
3.4.2	CNN-Metric	45
3.5	Experiments	47
3.5.1	Evaluation Benckmarks	47
3.5.2	Local Labeling Results	48
3.5.3	Evaluation of Global Features	52
3.5.4	Global Labeling Results	53
3.5.5	Integration Labeling Results	56
3.5.6	Analysis of Local and Global Beliefs	57
3.6	Conclusion	60

4	DAG-Recurrent Neural Networks	61
4.1	Introduction	61
4.2	Segmentation Network	63
4.3	Context aggregation via convolution kernels	64
4.4	DAG-Recurrent Neural Networks	65
4.4.1	Chain-RNN Revisited	66
4.4.2	DAG-RNN	66
4.4.3	Decomposition	69
4.4.4	Segmentation network with DAG-RNN	70
4.5	Attention to Rare Classes	71
4.6	Implementation Details	73
4.7	Main Results	75
4.8	Detailed study	77
4.8.1	Comparison with FCN	77
4.8.2	Comparison with Dilated FCN	79
4.8.3	Comparison with Fully Connected CRF	81
4.8.4	Ablation Study	84
4.9	Discussion	86
4.9.1	DAG-RNN on PASCAL VOC	86
4.9.2	Context vanishing in DAG-RNN	88
4.10	Conclusion	89
5	IFCN: towards achieving robust high-level and low-level scene parsing	91
5.1	Introduction	91
5.2	Segmentation Networks	94
5.2.1	Dilation or Skip?	94
5.2.2	Dense skip: a more effective approach to retain low-level information	97
5.2.3	Context Aggregation for high-level features	98

5.2.4	IFCN	101
5.3	Implementation Details	102
5.4	Architecture Evaluation on ADE20K	104
5.4.1	Which skip layers should be included?	105
5.4.2	Can the proposed architecture work for better pre-trained CNN? .	105
5.5	Results on segmentation benchmarks	106
5.5.1	Evaluation datasets	106
5.5.2	Result Summary	109
5.6	Conclusion and Discussion	110
6	Conclusion and future work	111
6.1	Conclusion	111
6.2	Future works	114
	Publications	116
	Appendix	119
	References	119

Abstract

In this thesis, we address the challenging task of scene segmentation, which generally refers to parsing a scene image into a set of coherent semantic regions. Scene segmentation requires multi-level scene understanding ranging from high-level semantic recognition to low-level boundary detection. Therefore, high-level contextual modeling as well as low-level information incorporation are both important in the functionality of scene segmentation models. In addition, semantic classes appear disproportionally in scene images, which further complicates the learning process of segmentation models.

In this thesis, we focus on addressing these challenges inherent to scene segmentation. More specifically, we mainly explore and discuss how effective high-level context should be incorporated in scene semantic segmentation. In the mean time, we discuss how to effectively retain low-level information that are essential in accurate low-level semantic parsing. Besides, we discuss and explore the issue of class frequency imbalance, which is severe and significant in scene semantic segmentation. Accordingly, we present three works, each of which explore and address parts of these challenges individually.

First of all, we adopt Patch-Convolutional Neural Network (Patch-CNN, which is trained from scratch with image patches) to be our parametric model for local patch analysis. Based on the occurrence frequency distribution of classes, an ensemble of CNNs (CNN-Ensemble) are learned, in which each CNN component is trained with different patch sampling strategy and each one focuses on learning different and complementary visual patterns. We notice that different CNN component performs significantly differently at parsing low-frequent semantic classes. Importantly, we observe a significant

performance boost of CNN-Ensemble for parsing low-frequent classes. Quantitatively, a significant **14%** and **8%** boost is achieved in terms of average class accuracy by CNN-Ensemble in comparison with single CNNs. Furthermore, we leverage the global scene semantics to alleviate the local ambiguity. The global scene constraint is practically estimated via an non-parametric framework. In the end, the integration of local and global beliefs gives rise to the class likelihood of pixels, based on which maximum marginal inference is performed to generate the label prediction maps. As our experiments demonstrate, the incorporation of global context significantly enhance the visual quality of prediction maps. Specifically, it improves approximately **5%** and **8%** in terms of overall pixel accuracy on SiftFlow and Barcelona dataset respectively. Even without any post-processing, the proposed algorithm achieves very competitive results on public scene segmentation benchmarks.

Secondly, we discuss how to effectively capture the rich contextual dependencies over image regions. Specifically, we propose Directed Acyclic Graph - Recurrent Neural Networks (DAG-RNN) to perform context aggregation over locally connected feature maps. More specifically, DAG-RNN is placed on top of pre-trained CNN (feature extractor) to embed context into local features so that their representative capability can be enhanced. In comparison with plain CNN (as in Fully Convolutioal Networks - FCN), DAG-RNN is empirically found to be significantly more effective at aggregating context. Therefore, DAG-RNN demonstrates noticeably performance superiority over FCNs on scene segmentation. Besides, DAG-RNN entails dramatically less parameters as well as demands fewer computation operations, which makes DAG-RNN more favorable to be potentially applied on resource- constrained embedded devices. Overall, the incorporation of DAG-RNN brings in a significant **3%** and **4%** IOU (Intersection of Union) boost over competitive FCN baseline on SiftFlow and Pascal-Context dataset respectively. Meanwhile, in order to address the imbalanced class occurrence frequency distribution, we propose a novel class-weighted loss to train the segmentation network . The loss distributes reasonably higher attention weights to infrequent classes during network training, which

is essential to boost their parsing performance. In our ablation experiments, we observe a significant **3.2%** ACA and **1.2%** IOU disparity between DAG-RNN trained with and without the proposed loss, which clearly demonstrates its effectiveness of improving segmentation performance for rare classes. We evaluate our segmentation network on three challenging public scene segmentation benchmarks: Sift Flow, Pascal-Context and COCO Stuff. On top of them, we achieve state-of-the-art segmentation performance.

Finally, considering that scene segmentation demands multi-level visual understanding ranging from low-level (e.g. boundary detection) to high-level (e.g. general object recognition). We firstly propose and place a convolutional context network (CCN) on top of pre-trained CNNs, which is used to aggregate contexts for high-level feature maps. We also discuss the limitations of current parameterization of skip layers, which are used to retain low-level information. By slightly modifying the parametrization of skip layers, we demonstrate that segmentation network with our skip layers delivers a very promising network architecture. In order to retain as much detailed low-level information as possible from pre-trained CNN, we introduce “dense skip” network architecture. We name our segmentation network improved as fully convolutional network (IFCN) based on its significantly enhanced structure over FCN. We carry out careful ablation studies to justify each contribution individually. Overall, IFCN outperforms the competitive FCN baseline by a significant margin of **7%** and **8.8%** IOU on the challenging ImageNet (ADE20k) and Pascal-Context dataset respectively. We also compare their qualitative segmentation maps, and demonstrates how IFCN achieves robust high-level as well as low-level parsing results. Without bells and whistles, IFCN achieves state-of-the-arts on ADE20K, Pascal Context and Pascal VOC 2012 segmentation datasets.

In summary, we have explored three key aspects that are of great importance to enhance scene segmentation performance: incorporating high-level context (contextual modelling), retaining low-level information as well as boosting recognition performance for rare classes. Step by step, we come up with a state-of-the-art segmentation network - IFCN that integrates the best configuration (of these three aspects) to our best research endeavours.

List of Figures

1.1	Three representative datasets for scene segmentation. In the demonstration, we show input images and their expected output label maps (ground truth). As clearly shown, images are densely and richly annotated, wherein many small-size or thin-shaped objects are commonly parsed out.	3
2.1	An image is processed by a typical segmentation network that subsumes three key functional modules and a optional post-processing module. . .	11
2.2	Examples of human annotations on object segmentation and scene segmentation datasets. Input images in the 1 st row are sparsely annotated in 2 nd row (object segmentation, i.e. Pascal VOC 2010 dataset), and their dense and rich semantic annotations are displayed in 3 rd row (scene segmentation, i.e. Pascal Context dataset).	13
3.1	Motivation of our integration model: the parametric model can distinguish visually different pixels very well, but get confused for pixels that are visually similar in local context. However, the local ambiguities can be easily eliminated as long as the correct global scene semantics is revealed. A more consistent labeling result can be achieved by integrating their beliefs. The figure is best viewed in color.	32

3.2	Framework of our integration model: the parametric CNNs are responsible for emitting the local beliefs, and the non-parametric model is practically used to output the global beliefs. The integration of local and global beliefs gives rise to the un-normalized class likelihood of pixels. The modules painted in yellow represents parametric models (CNN and Metrics). . . .	36
3.3	The first graphic shows an example of the class frequency distribution in natural scene images, through which we can observe that large percentage of pixels belong to very small number of frequent classes. The second figure delineates the class-wise accuracy curves for CNN-M. The experimental statistics are based on the SiftFlow dataset [1].	40
3.4	Graphical illustration of the effect of our large margin based metrics. Due to the highly imbalanced data distribution, the nearest neighbors of the testing feature (triangle) are dominated by imposter classes (rectangle). After the metric transformation, the imposters stay far away from it. Thus, their contribution is significantly attenuated in the global belief estimation.	46
3.5	Visualization of the learned convolution filters for the first layer of CNN-CS, CNN-TCS, CNN-GS and CNN-HS respectively. They are trained on the Barcelona datasets. Note the visual difference among the learned filters of different CNN-M. The figure is best viewed in color.	49
3.6	Qualitative labeling examples on the SiftFlow dataset (best viewed in color). In each row, we show input images, top three nearest exemplar images, local labeling maps (output by the parametric model - CNN-Ensemble), global labeling maps (output by the non-parametric model), final label prediction maps (output by the integration model) and their ground truth maps respectively. The numbers outside and inside parentheses are global pixel accuracy (GPA) and average class accuracy (ACA) respectively. The last row shows an example, where the <i>local ambiguity</i> cannot be removed when their aggregated global features fail to reveal the true scene semantics.	58

4.1	Our basic segmentation network essentially consists of three temporal-consecutive functional modules: local representation, context aggregation and feature map upsampling. Below, an example of the proposed segmentation network is demonstrated that subsumes convolutional network, DAG-RNN and deconvolution network. C denotes the cardinality of semantic classes considered in the target dataset. Shortcut lines (connections) symbolize that the outcoming feature maps (e.g. $28 \times 28 \times 512$) are firstly locally predicted and then summed with preceding (upsampled) predictions.	63
4.2	Illustration of context aggregation. (a), feature tensor (with size of $8 \times 8 \times d$) for the given scene image, and each square denotes one feature vector in the feature tensor; (b), convolution kernel (with the size of $7 \times 7 \times d \times m$) is used to aggregate context for local features. The two dashed squares symbolize the field of views of two corresponding local features (highlighted with blue/magenta color); (c), local features are connected in an 8-neighborhood UCG, under which any pair of local features can be mutually communicable; and (d), one of its induced DAG in the southeastern (SE) direction. Context aggregation is achieved by propagating local context based on such DAG topology.	65
4.3	An example illustrates the decomposition of an UCG to four complimentary DAGs: $\mathcal{U} = \{\mathcal{G}_1, \dots, \mathcal{G}_4\}$. Note that any vertex pair can still be mutually communicable in the resulting set of DAGs. For example, local information of the yellow vertex can be routed to magenta vertex via $\mathcal{G}_1$, meanwhile yellow vertex can glean features from magenta vertex via propagating path in $\mathcal{G}_4$	69
4.4	Graphical visualization of the class frequencies (left) and weights (right) on the sift Flow datasets [1]. The classes are sorted in the descending order based on their occurrence frequencies in training images.	71

4.5	Detailed class-wise accuracy comparisons on Pascal Context dataset. The classes are presented in the descending order based on their occurrence frequencies, and rare classes are highlighted with blue color.	82
4.6	(Best viewed in electronic version with zooming in.) Example of qualitative parsing results on Pascal Context dataset. We present six images (from left to right, top to down) in a block which represent input image, human annotation, label maps predicted by baseline FCN, baseline FCN + CRF [41], DAG-RNN and DAG-RNN + CRF respectively. The examples in the first row demonstrate the case in which recognition accuracies of “thing” (i.e. tv monitor, bird, train and boat) are phenomenally increased after DAG-RNN is used. The second-row example show that DAG-RNN improves the parsing performance for “stuff” (i.e. floor, water, wall, road, etc). The third-row examples present some interesting cases, where wrong predictions (such as ‘ground’ or ‘dog’) also make sense.	87
4.7	Two UCGs (with 4, 8 neighborhood system) and their induced DAGs in the northwestern (NW) direction. In comparison with DAG(4), DAG(8) enables information to be propagated in shorter paths, which is critical to prevent the long-range information from vanishing. As an example, the length of propagation path from v_9 to v_1 in $\mathcal{G}_{nw}^8$ is halved to that in $\mathcal{G}_{nw}^4$ ($4 \rightarrow 2$ steps) due to the shortcut path.	89
5.1	(Best viewed in color) Semantic segmentation demands robust high-level as well as low-level parsing. IFCN outperforms FCN both in high-level smoothing / recognition (“train” and “ground” in the first and third example) and low-level boundary localization (e.g. bird legs in second example).	92

5.2	Skip layers in segmentation networks. The right figure depict its detailed computational blocks. Mathematically, $\mathbb{H} = \mathbb{H}' + S(\mathbb{F}, \Theta)$, where $\mathbb{F}$ and $\mathbb{H}$ denotes input feature map (with the dimensionality of $height \times width \times \#channels$) and its class-likelihood map respectively, $S(\cdot, \cdot)$ is the computational function of skip layer and Θ is their learnable parameters. $\mathbb{H}'$ represents class-likelihood maps generated from other branches.	96
5.3	convolutional context network (CCN) is a convolutional network with dense skip layers. Contextual field (orange coverage in $\mathbb{F}'$) of feature maps are progressively expanded. Note that features in $\mathbb{F}'$ incorporates all elements in $\mathbb{F}$	99
5.4	Network architecture of IFCN-xs. Similar to the demonstration of FCN-xs, feature maps are shown as grids that reveal relative spatial coarseness. IFCN-4s and IFCN-2s can be trivially inferred from the above architecture demonstration. The black vertical line denotes the conv layer in the pre-trained CNN (VGG-16). In order to save space, some feature maps are not displayed. Note that the detailed architecture of convolutional context network (CCN) can be retrieved in Figure 5.3.	102
5.5	Network architecture comparison. These networks differentiate each other from two aspects: (1), how to adapt pre-trained CNN to retain detailed spatial information; (2), how to aggregate context for high-level feature maps.	103
5.6	(Best viewed in color with 300% zooming-in) Qualitative comparisons of parsing maps. DAG- RNN [76] is very good at enforcing high-level consistency (first three examples). Due to the adopted “dilation” architecture, DeepLab-v2 [22] and DilatedNet [34] perform well at localizing boundaries (4 th and 5 th example). IFCN performs equally well in low-level as well as high-level parsing.	108

List of Tables

3.1	Quantitative Performance of different CNNs. Details of each method are elaborated in the text.	50
3.2	Average genuine matching percentage among their K-nearest neighbors for different global features. GT is the semantic feature pooled from ground truth label maps.	52
3.3	Quantitative performance of different methods on the SiftFlow datasets. .	55
3.4	Quantitative performance of different methods on the Barcelona dataset.	56
3.5	Per-class accuracy comparison of different models on the SiftFlow dataset. The statistics of class frequency is obtained in test images.	59
4.1	Performance comparison on the Pascal Context dataset (59 classes). For fair comparison, we only present methods that use VGG-16 [13] as feature extractor. DAG-RNN (with CRF [41]) achieves the best results.	76
4.2	Performance comparison on the Sift Flow dataset. DAG-RNN achieves very competitive performance that is on par with state-of-the-arts. . . .	77
4.3	Performance comparison on COCO Stuff dataset (171 classes). DAG-RNN significantly outperforms the popular networks - FCN and DeepLab. . .	78

4.4	Quantitative result comparisons on scene segmentation benchmarks. A typical input image has the resolution of 256 x 256 and 375 x 500 pixels for Sift Flow and Pascal Context datasets respectively. To facilitate study, a light model (hidden dimension of DAG-RNN is 128) is adopted in the controlled experiments. Therefore, there is a slight discrepancy between the performance listed here and the reported ones (DAG-RNN) in Table 4.1, 4.2.	80
4.5	Detailed comparison of Context Aggregation Module (CAM) over different segmentation networks. FLOPs denote the count of floating-point operations. the GPU / CPU flag symbolizes that CAM of these comparing networks are executed on the corresponding devices. The batch-based inference testing takes input as 448×448 images (re-sized and zero padded). Feature maps that are fed to CAM have the spatial resolution of 56×56 and 14×14 for DilatedNet and the rest networks respectively.	81
4.6	Quantitative result comparison on Pascal Context dataset (59 classes, including 22 (37) frequent (rare) classes). The introduced class-weighted loss delivers more attention to rare classes during network training. In consequence, the model (trained with weighted loss) performs significantly better at recognizing rare classes, while simultaneously working very well at parsing frequent classes.	85
4.7	Quantitative result comparison of DAG-RNN with different number of DAGs. Note that the segmentation network does not have context aggregation module in the case where no DAGs are used in DAG-RNN. . . .	86

5.1	Two approaches - “Dilation” and “Skip” - are used to adapt pre-trained CNN (in this case, VGG-16 [13]). Results of their corresponding networks are evaluated on ADE20K dataset [12]. Networks are trained under the same condition (details are elaborated in Section 5.3) except that FCN-8s(Orig) is trained without class-weighted loss [76]. Speed refers to the inference time of segmentation networks on a single NVIDIA Titan X. . .	95
5.2	Parsing performance comparison between “sparse skip” and “dense skip” network architectures on ADE20K dataset [12]. Here skip refers to our new parameterized skip layers.	98
5.3	Segmentation networks are adapted to “our skip” architecture, and they differentiate each other in terms of context aggregation. Results are evaluated on ADE20K dataset.	101
5.4	Parsing performance of different networks (c.f. Fig 5.5) on ADE20K dataset.	104
5.5	Performance comparisons of different IFCN architectures on ADE20K dataset. For efficiency reasons, the hidden dimension of CCN is reduced to 512, thus a slight performance discrepancy is observed when it is compared with that in Table 5.4.	105
5.6	Performance (GPA, ACA, IOU) of different network architectures (with different pre-trained CNNs) on ADE20K dataset.	106
5.7	Pascal Context validation accuracies. For fair comparisons, we only include methods that use VGG-16 as base network.	107
5.8	SUN-RGBD (37 classes) testing accuracies (%). We only use RGB modality in our experiments.	107
5.9	Pascal VOC 2012 testing accuracies. For fair comparisons, we only include methods that use VGG-16 as base network and does not use auxiliary segmentation masks for pre-training.	109

List of Abbreviation

ACA	Average Class-wise Accuracy
ASPP	Atrous Spatial pyramid pooling
BPTT	Back Propagation Through Time
CAM	Context Aggregation Module
CCN	Convolutional Context Network
CNN	Convolutional Neural Network
DAG	Directed Acyclic Graph
DAG-RNN	Directed Acyclic Graph - Recurrent Neural Network
DCNN	Deep Convolutional Neural Network
FCN	Fully Convolutional Network
GPA	Global Pixel Accuracy
GPU	Graphical Processing Unit
HOG	Histogram of Oriented Gradients
IFCN	Improved Fully Convolutional Network
IOU	mean Intersection Over Union
LSTM	Long Short Term Memory
PGM	Probablistic Graphical Model
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SIFT	Scale Invariant Feature Transform
SPM	Spatial Pyramid Matching
SVM	Support Vector Machine
UCG	Undirected Cyclic Graph

List of Notation

$(\cdot)^T$	Matrix or vector transpose
$\circ$	Hadamard product
$\sqrt{\cdot}$	Square root
$\exp(\cdot)$	Exponential function
$\log(\cdot)$	Logarithmic function
$\mathbf{x} \in \mathbb{R}^D$	A D -dimensional vector
$\ \mathbf{x} - \mathbf{y}\ _2$	Euclidian distance between vectors $\mathbf{x}$ and $\mathbf{y}$
$\ \mathbf{x}\ _2$	L2 norm of the vector $\mathbf{x}$
$\mathbf{X} \in \mathbb{R}^{D_1 \times D_2}$	A matrix $\mathbf{X}$ of size $D_1 \times D_2$
$\ \mathbf{X}\ ^2$	Frobenius norm of the matrix $\mathbf{X}$
$\mathcal{X} \in \mathbb{R}^{h \times w \times c}$	A tensor $\mathcal{X}$ of size $h \times w \times c$

Chapter 1

Introduction

Scene segmentation (also termed as scene labeling, scene parsing, semantic segmentation) builds a bridge towards deeper scene understanding. It refers to decomposing a scene image into a set of coherent semantic regions. In other words, it associates each pixel with one of pre-defined semantic classes. In comparison with shallow scene analysis, such as scene classification which simply categorize the whole image as a pre-defined scene type (e.g. outdoor coast, outdoor highway, indoor kitchen, etc), scene segmentation provides richer clues, based on which many useful and interesting interpretations can be implicitly inferred, such as its global scene type, the locations of its constituent objects, the possible interactions between different objects, etc.

Based on the shapeness of semantic classes, we roughly categorize these classes to ‘stuff’ or ‘thing’ [1][2][3]. To elaborate, ‘stuff’ usually refers to those classes without explicit shapes, such as ‘sky’, ‘road’ and ‘water’, etc. On the contrary, ‘thing’ generally refers to object classes that have a discriminative (may be deformable) shape, such as ‘person’, ‘boat’ and ‘cat’, etc. In general, “thing” classes simply refer to objects. Thus, throughout this thesis, object classes have the same semantic meaning of “thing” classes unless specifically explained. In general, ‘thing’ pixels in real world images exhibit huge appearance variances due to scale, illumination and pose changes. In the mean time, ‘stuff’ pixels with different semantic meaning can be visually similar in a close-up view.

For example, the appearance of a chihuahua (a dog breed) playing outdoors can be significantly different from that of a husky (another dog breed) laying on the floor, although their occupying regions belong to the same ‘dog’ (thing) class. Meanwhile in some cases, local ‘sea’ regions can be visually indistinguishable from ‘road’ regions if they are placed out of context, although their belonging semantic classes are totally distinct. Therefore, local classification over image pixels (or patches) can be quite challenging. In addition, class frequency distribution is highly imbalanced in natural scene images: more than 80% pixels in the images belong to only a few number of semantic classes. Thus, the learned classification model is usually hard to generalize well to infrequent classes. Overall, these issues pose scene segmentation to be one of the most challenging problems in computer vision.

Scene segmentation provides an essential module in many real-world application including auto-driving or assistant-driving (ADAS)[4], Virtual reality (VR), Augmented reality (AR). Meanwhile, robust scene parsing itself can be potentially used in image editing as well. Importantly, as scene segmentation offers a very deep semantic understanding to the given image, it therefore provides very precious semantic information for other application-based research topics such as multiple-object tracking [5][6], semantic SLAM [7][8], and textual description of images [9][10], just to name a few. Thus, scene segmentation is a fundamental topic that is also worth exploring from industrial prospective.

1.1 Challenges and Motivations

In order to precisely parse a given scene image, a good segmentation network needs to address the following challenges. In summary,

- (i) In general, “stuff” classes and large-size “thing” (object) classes occupy a large percentage of pixels in the image. Thus, visual appearance can be significantly

SiftFlow [1]

Pascal Context [11]

ADE20K(ImageNet Scene Parsing Challenge 2016) [12]

Figure 1.1: Three representative datasets for scene segmentation. In the demonstration, we show input images and their expected output label maps (ground truth). As clearly shown, images are densely and richly annotated, wherein many small-size or thin-shaped objects are commonly parsed out.

inconsistent across different parts of the same instance. Take the 1st (SiftFlow) image in Figure 1.1 as an example, the appearance of upper regions of “mountain” looks significantly different from that of lower-regions. Meanwhile, different parts of the “bed” in the 1st (ADE20K) image also looks visually different. Thus, it is practically challenging to generate a consistent class prediction to those visually inconsistent regions.

- (ii) In complicated scenes, small-size objects are predominant (due to its small scale, its shape). These small-size objects usually appear in low-resolution (c.f. small person, lights in Figure 1.1), so it is challenging to represent them in a robust and discriminative form. Meanwhile, current post-processing methods (e.g. Fully connected Conditional Random Field - CRF) usually smooth out those small-size region predictions, thus it is extremely challenging to parse out these pixels. Meanwhile, accurately localizing region boundaries are essential in robust scene segmentation. Unfortunately, it’s of great difficulty to precisely detect those class boundaries.
- (iii) As mentioned earlier, semantic classes appear severely disproportionally in scene images. Due to the scarcity of rare-class training examples, it is hard for segmentation models to learn robust and generalized representation for rare classes. Specifically, the training loss from frequent classes dominates the total training loss in each training batch. Thus, rare classes actually receive very few supervision signals that are essential to learn generalized representation for them. This highly data imbalance will make the segmentation model prioritize predicting frequent classes (e.g. sky, road, etc), which is not desirable in image understanding.

The above mentioned challenges are inherent to scene segmentation, and they motivate us to explore the following directions (aspects) to develop the segmentation algorithms specifically in this thesis.

- (i) **High-level Context Aggregation** is to enhance the high-level feature representation of image regions ¹, which is important to enforce high-level (higher-order) label consistency. It is aimed to boost the segmentation performance of “stuff” classes as well as large-size “thing” (object) classes. In this thesis, we are dedicated to exploring how to build effective contextual model to phenomenally enhance the high-level parsing performance of segmentation models.
- (ii) **Low-level Information Retention** is to incorporate precious low-level and local image cues into local parsing. It is aimed to improve the parsing performance of small-size objects as well as to enhance the recognition accuracy of region boundaries. In this thesis, we discuss how to incorporate such information in deep neural networks to boost the performance of low-level scene parsing.
- (iii) **Addressing Severe Data Imbalance** is to alleviate the negative effect of scarcity of infrequent classes. It is aimed to improve the recognition accuracies of rare classes without significantly sacrificing the segmentation performance of frequent classes. In this thesis, we discuss how to mitigate the severe data imbalance in training batches so that rare-classes substantially contribute to the training loss.

1.2 Proposed Segmentation Models

In this thesis, we present three scene segmentation models , each of which address several challenges discussed earlier. We are to briefly introduce each segmentation model individually.

Our first segmentation model is based on Patch-Convolutional Neural Network (Patch-CNN), which is trained from the scratch with image patches. Furthermore, we propose to utilize the global context (scene semantics) ² to refine the local parsing results (usually

¹Note that we already have a feature extractor (different convolutional neural networks are used in this thesis), based on which we generate the corresponding local features for image regions.

²Global context is significant to mitigate local ambiguity, which refers to the situation that pixels or patches are visually indistinguishable in a local view (i.e. out of context).

noisy) so that high-level label consistency is achieved. Specifically, a global scene contextual constraint is implicitly enforced to allow more reliable local classification. Such a global scene constraint is practically modeled in a non-parametric framework by transferring the class dependencies and priors from its global similar exemplar images. In summary, the segmentation model integrates the strengths of patch-CNN as well as global context. This segmentation model achieves very competitive segmentation performance on public scene segmentation datasets including SiftFlow [1] and Barcelona [2]. Meanwhile, we also discuss how patch sampling influences the per-class recognition performance of patch-CNN.

Our second segmentation model is based on significantly stronger feature extractor - deep CNN (i.e. VGG-16 [13]) pre-trained on ImageNet datasets [14][15]. Then, DAG-RNN is proposed and placed on top of pre-trained CNN to capture the rich contextual dependencies over image regions so that representative capability of feature maps can be significantly enhanced. In detail, DAG-RNN, a generalization of RNNs [16][17], is to explicitly pass local context based on the directed graphical structure. By doing this, contexts are explicitly propagated and encoded into feature maps. Overall, this segmentation model integrates FCN and DAG-RNN in an end-to-end trainable network architecture. Importantly, it achieves state-of-the-art segmentation performance in public datasets including SiftFlow [1], Pascal Context [11] and COCO-Stuff [18]. It's important to note that this segmentation network is trained with a novel proposed loss which is to distribute higher attention weight for infrequent classes and thus to significantly boost the segmentation accuracies for those infrequent classes.

Our third segmentation model is also an end-to-end trainable network architecture³ that is based on pre-trained deep CNNs (e.g. VGG-16 [13], ResNet [19]). First, a convolutional context network (CCN) is proposed and placed on top of pre-trained CNNs, which is used to aggregate contexts for high-level feature maps so that robust high-level

³This segmentation network (model) is also trained with the loss that has been used in training DAG-RNN.

parsing can be achieved. Furthermore, “dense skip” architecture is introduced to retain rich set of low-level information from pre-trained CNN, which is essential to improve low-level parsing performance. The resulting segmentation network is fully convolutional, and it has dense skip layers from pre-trained CNN and CCN. Thus, we name it as improved FCN (IFCN) based on its enhanced structure over FCN. Overall, IFCN performs equally well on high-level as well as low-level parsing. In practice, it achieves state-of-the-art performance on standard semantic segmentation datasets including ADE20K [12], Pascal Context [11] and VOC 2012 [20].

Overall, all of these proposed models are to address the proposed research challenges in Section 1.1 that are inherent to scene segmentation. Their segmentation performance is progressively advanced, which aligns with the research direction in semantic segmentation community as well. More importantly, both DAG-RNN and IFCN achieve state-of-the-art on public scene segmentation benchmarks.

1.3 Contributions

In summary, the three segmentation models that are proposed in this thesis make the following contributions:

- (i) We propose three different context aggregation modules to enforce high-level parsing consistency. Firstly, we explore to use global context to refine local parsing results in an integration model, and the incorporation of global semantics brings in a phenomenal **5%** and **8%** improvement over overall pixel accuracy on SiftFlow [1] and Barcelona dataset [2] respectively. Secondly, we propose DAG-RNN to model the rich contextual dependencies over image regions, and our DAG-RNN outperforms the competitive FCN (without high-level context aggregation module) by a significant margin of **3%** and **4%** Intersection of Union (IOU) on SiftFlow [1] and Pascal-Context [11] dataset respectively. Finally, we introduce a high-capacity

context aggregation network - Convolutional Context Network (CCN), and the incorporation of our CCN brings in a significant **7%** IOU boost to Fully Convolutional Network (FCN) on the challenging ADE20K dataset [12].

- (ii) We propose two different effective methods to address the severe class imbalance issue under different frameworks. First, we explore how patch sampling influences the per-class parsing performance in Patch-Convolutional Neural Network (Patch-CNN, trained from the scratch with image patches). We also discuss which patch sampling strategy achieves the best per-class recognition performance tradeoff. We propose to combine different complementary CNN components trained with different sampling strategy, and this practice leads to a significant **14%** and **8%** improvement in terms of average class accuracy (ACA) on SiftFlow [1] and Barcelona dataset [2] respectively. Secondly, we propose a novel loss function to distribute higher learning rates (i.e. attention weights) for rare-classes in the end-to-end learning paradigm. We observe a significant **3.2%** ACA improvement when the proposed loss is engaged in DAG-RNN training on challenging Pascal-Context dataset [11].
- (iii) We also investigate how to effectively incorporate low-level image cues to improve scene segmentation performance. We firstly analyze the limitations of current parametrization of skip layers.⁴ By slightly modifying its parametrization, the performance of the same segmentation networks (e.g. FCN [21] and DeepLab [22]) can be significantly boosted by **1.4%** IOU on ADE20K dataset [12]. Moreover, we further propose the “dense skip” architecture to adapt the pre-trained feature extractor (CNN) so that lower-level features are effectively leveraged into local predictions. This architecture design leads to another **1.5%** IOU improvement on ADE20K dataset [12].

⁴Skip layers are used to explicitly incorporate lower-level feature maps into semantic segmentation, which enables the resulting segmentation networks (e.g. FCN-8s [21] vs FCN-32s) to generate significantly more detailed predictions.

- (iv) We propose two end-to-end trainable segmentation networks (DAG-RNN and IFCN) that achieve state-of-the-art segmentation performance on challenging scene segmentation benchmarks including Pascal-Context [11], COCO-Stuff [18] and ADE20K dataset [12].

1.4 Thesis Outline

The rest of the thesis will be organized as follows: In Chapter 2, we review, compare and discuss related literatures in depth. As follows, we elaborate our proposed segmentation model. Chapter 3 presents the motivation, formulation as well as experiments of our segmentation model that integrates patch-CNN with global context. Then, the motivation, computational model as well as detailed ablation studies of DAG-RNN are elaborated in Chapter 4. As follows, Chapter 5 demonstrates the detailed model of IFCN as well as its evaluation and ablation studies on public scene segmentation benchmarks. Finally, in Chapter 6, we conclude and prospect this thesis by retrospecting its contributions as well as discussing interesting future works.

Chapter 2

Literature Review

In this chapter, we firstly discuss and compare the differences between two similar tasks – object segmentation and scene segmentation, as both tasks are usually referred to as semantic segmentation in informal scenarios. In this thesis, we advocate to differentiate these tasks, as we strongly believe that scene segmentation has significantly distinct research challenges (from object segmentation) that needs in-depth research endeavours. Accordingly, we review and discuss previous literatures that address these two tasks. Next, we review, discuss and compare previous works that are relevant to our research from different perspectives. We also illustrate how our contributions technically differ from previous works. In the end, we systematically summarize the contributions of this thesis in position with previous works.

2.1 Object segmentation vs Scene segmentation

2.1.1 Object Segmentation

Object segmentation refers to parsing out the objects of interest from the image, and it is largely popularized by Pascal VOC segmentation challenge [20]. Examples of expected output maps are presented in Figure. 2.2. Beyond locating the position or figuring out the scale of objects in detection task, object segmentation needs to further understand the silhouette of detected objects. In more detail, it's essential to draw the exact image regions (rather than a tight bounding box) where objects appear in object segmentation.

Figure 2.1: An image is processed by a typical segmentation network that subsumes three key functional modules and an optional post-processing module.

A typical state-of-the-art segmentation network ¹ generally follows the architecture design illustrated in Figure. 2.1, although its implementation can be different. Researchers are dedicated to crafting better computational model for each functional module (sub-networks) in order to improve the overall performance of segmentation networks. In accordance, the performance of object segmentation has been steadily and dramatically improved every year. In one aspect, such performance enhancement should be largely attributed to the magnificent progress of pre-trained CNN, which is simply adapted to be the feature extractor in segmentation networks. In this regard, local features are less sensitive to the image nuances (e.g. scale, illumination variations, etc), so their unary predictions are more consistent and reliable. This progress in the core of improving better model on large-scale image classification datasets (e.g. ImageNet [14] [15]) by training deeper or more complicated networks ² such as AlexNet [23], GoogLeNet [24], VGG-16 [13], ResNet [19], Inception-ResNet [25], ResNext [26], DenseNet [27], Xception [28] etc. Meanwhile, many researchers are dedicated to developing novel computational layers that are able to effectively encode informative context into local feature maps. This research direction plays a significant role to enhance the visual quality of prediction label maps as well as to boost the segmentation accuracy. Representative works such as DPN [29], CRF-CNN [30], ReSeg [31], DeepLab-v2 [22], RecursiveNet [32], ParseNet [33], Dilated-Net [34], RefineNet [35], PSPNet [36] formulated their computational layers to achieve

¹We only consider and review literatures that adopting deep neural networks (i.e. methods in post-CNN era) to perform object segmentation, which is more related to this thesis.

²We are not going to review the advancement of pre-trained CNN here, as this is not the main focus of this thesis.

effective context aggregation³, and they are experimentally demonstrated to significantly benefit the segmentation accuracy on Pascal VOC benchmarks. In addition, research endeavours have also been devoted to recovering the detailed spatial information by either learning a deep decoder network [37][38] [39] [40] or applying a disjoint post-processing module such as fully connected CRF [41] [42][43].⁴ These techniques have collectively pushed the segmentation performance saturating on Pascal VOC benchmarks.⁵ The steady progress also calls for the unveiling of new and more challenging benchmarks (e.g. Microsoft COCO [44] dataset).

2.1.2 Scene Segmentation

Scene segmentation, on the other hand, requires to semantically parse every pixel in the image. In Figure 2.2, we demonstrate the disparity of annotations between scene segmentation with object segmentation, which largely differentiate these two tasks. Although scene segmentation shares substantial similarity with object segmentation and the techniques that advance object segmentation can be transferred to boost the scene parsing performance, it has specific characteristics and challenges and few literatures address them carefully. In summary,

- (i) Images in scene segmentation are usually annotated with richer category annotations. For example, each pixel belongs to one of 21 and 59 categories for images from Pascal VOC [20] and Context datasets [11] respectively. Not surprisingly, semantic classes are correlated in a more complicated manner (e.g. stuff-stuff, stuff-object and object-object correlations).
- (ii) More objects (including small-size , thin-shaped objects) are segmented out in scene segmentation, and their boundaries are significantly more challenging to be detected.

³Their detailed architecture description and how our technical contributions differ from them will be elaborated in Section 2.2.5

⁴The detailed literature review of this part will be presented in Section 2.2.7

⁵ More than 85% mean IOU has been achieved by state-of-the-art segmentation models.

Figure 2.2: Examples of human annotations on object segmentation and scene segmentation datasets. Input images in the 1st row are sparsely annotated in 2nd row (object segmentation, i.e. Pascal VOC 2010 dataset), and their dense and rich semantic annotations are displayed in 3rd row (scene segmentation, i.e. Pascal Context dataset).

- (iii) Semantic classes appear severely more disproportionately in fully-annotated scene images, and segmenting rare classes is significantly more difficult.

In align with these challenge differences, we technically discuss how their corresponding research methods differ as follows.

Contextual reasoning. In contrast to object segmentation, contextual reasoning is more demanding in scene segmentation. More specifically, in addition to capture object-object dependencies (as advocated by the challenging Microsoft COCO benchmarks [44]), a scene segmentation model should also be able to model stuff-stuff (e.g. ‘sky’-‘road’, ‘mountain’-‘sea’, etc) as well as stuff-object (‘road’-‘person’, ‘sea’-‘boat’, etc) interac-

tions. In the mean time, densely-annotated scene images are generally full of small-size objects (due to thin-shape, small-scale, etc), contextual reasoning is essential to recover the identity for these objects as well. As Section 2.1.1 elaborates, current trends of contextual modeling under the neural network framework focuses more on object-object interaction [30]. In this thesis, we explore how to effectively capture the rich contextual dependencies over local features, which is expected to model complex correlations among semantic classes in scene segmentation. The incorporation of effective contextual modelling leads to our models achieving very impressive results on public scene segmentation benchmarks. In detail, we propose three modules to encode contextual information into segmentation models either by late fusion of global beliefs (c.f. Chapter 3) or formulating new computational layers to aggregate context for feature maps (DAG-RNN (c.f. Chapter 4) and Convolutional Context Network - CCN (c.f. Chapter5)).

Complex boundary localization. In scene segmentation, many objects (regardless of their sizes, shapes, etc) as well as textureless ‘stuff’ classes are semantically annotated. Thus, boundary regions occupy more percentage of pixels compared to that in object segmentation. In order to obtain high-quality prediction maps, it’s essential to localize these complex boundaries accurately for segmentation models. In general, previous literatures [42] [43] perform a contextual post-processing step to refine object boundary delineation. Instead in deep neural network frameworks, many researchers explore retaining the essential low-level image cues via adapting the local representation module (i.e. pre-trained CNN) . For example, the seminal work - Fully Convolutional Network (FCN) [21] - demonstrates that more detailed parsing maps can be generated as lower-level feature maps are incorporated into predictions. From the perspective of architecture designs, this is achieved by adding skip layers from lower-level feature maps of pre-trained CNN. On the other hand, DeepLab [22] and DilatedNet [34] [40][45] [46] directly remove some subsampling layers (and the following convolution layers are substituted with atrous convolution accordingly to keep the receptive field of feature maps roughly the same) in

pre-trained CNN to avoid the gradual loss of detailed spatial information. In this thesis, we propose “dense skip” architecture to retain rich set of low-level detailed information, which is shown to be more effective as well as more efficient than previous works.

Imbalanced class distribution. Due to rich semantic annotations in scene segmentation, classes appear more disproportionately comparing to that in object segmentation. In reality, few frequent semantic classes (e.g. sky, road, etc) occupy large percentage of pixels. For the rest majority classes, they are either in small-size or statistically rare. In general, there are few effective training examples for infrequent classes, which hinders segmentation networks to learn a good and general representation for them.

In the early CNN period, Farabet et al. [47] explores how the patch sampling influences the segmentation performances over different classes. In detail, they propose “class balanced” sampling, by which all classes appear equally frequently in the resultant training batch. By doing this, the parsing performance of rare classes is significantly boosted. However, the overall visual quality of prediction maps deteriorates dramatically due to the significant degradation of segmentation performance for dominant frequent classes. In this thesis, we also discuss this line of work, and we propose “hybrid” sampling to mitigate the severe scarcity of rare classes in training batches. We also demonstrate that it is an effective approach for the segmentation model to learn a good representation for both frequent and rare class pixels. Additionally, we propose complimentary sampling methods in the aim to maximize the segmentation performance for semantic classes with different occurrence frequencies. By combining these complementary models, we are able to achieve very impressive parsing performance tradeoff across different semantic classes.

On the other hand, Mostajabi et al. [48] and Caesar et al. [49] weighs each class according to its inverse frequency in the loss function, which achieves promising results on the object segmentation but fails on scene segmentation. Inspired by the success of “hybrid” sampling in [50], we propose a novel class-weighted loss that distributes reasonably higher attention for rare classes based on their rareness magnitude. We also

demonstrate that the network trained with the proposed class weighted loss performs significantly better at parsing rare classes without significantly harming the performance of frequent classes.

In summary, a scene segmentation network architecture generally demands more effective high-level contextual modelling as well as requires more efficient low-level information retention in contrast to object segmentation networks, although both network backbones are functionally the same (as in Figure 2.1). Additionally, research efforts are also required to address the significantly more severe class frequency imbalance in training scene segmentation networks.

2.2 Related works

In this section, we start reviewing, discussing and comparing with previous works that are related to our contributions from different perspectives.

2.2.1 Feature Learning

This research direction exploits extracting better features for classifying pixels/superpixels. Previously, low-level and mid-level hand-crafted features are designed to capture different image statistics, such as SIFT [51], HOG [52], SURF [53], ORB [54], DAISY [55] etc. These descriptors usually lack discriminative power and suffer from high dimensionality, thus limiting the efficacy of the full segmentation model. Recently, machine learning techniques are commonly used to learn discriminative features for various computer vision tasks [47, 56, 57, 58]. Fed with raw image pixels, these learning frameworks are able to learn compact and discriminative features for the task at hand, thus significantly improving the overall segmentation performance. Here, we review two popular machine learning approaches that are widely used in scene semantic segmentation.

Random Forest [59] [60] is a collection of multiple deep decision trees, whose leaf

nodes store the probability distribution (over prediction labels) of the input data.⁶ At each intermediate tree node, the incoming data will be routed to two child nodes based on the maximization over the information gain. A tree node will not be split if all the incoming data has the identical labels or the number of incoming data is below a pre-defined threshold. Due to the simplicity of split function as well as testing features in each intermediate tree node, it's extremely efficient to go through a deep decision tree. Thus, random forest has been widely used in pixel-wise image analysis, such as scene labeling [61] [62] [63], edge detection [64] [65]. For example, Shtton [61] pioneered using activations of leaf nodes in decision forest to represent each image patch, and they achieved very impressive results on early semantic segmentation tasks. Buló et al. [63] learned a more compact and stronger random forest by substituting the random split function with a stronger Neural Network. Kotschieder et al. [62] and Dollar et al. [65] formulated pixel labelling (scene labeling and edge detection) as a structured learning problem, and they successfully adopted random forest to learn such complex structure transformation. Without learning an explicit form of features for each image patch, random forest practically evaluates a huge number of informative features ranging from color, gradient, edge, etc.

Convolutional neural network (CNN) is a type of feed-forward neural network and it is widely adopted in pattern recognition tasks. Specifically in the domain of image analysis, the architecture of a convolutional neural network can be functionally decomposed to two different modules (sub-networks) (1) feature extraction, and (2) classifier. To be more specific, the beginning convolutional blocks⁷ can be regarded as feature extraction module, via which an image is represented as a high-dimensional features that are ready for semantic image analysis. Meanwhile, the following fully connected (**fc**) layers can thus be treated as a discriminative classifier, which is fed with the high-dimensional

⁶In scene segmentation scenario, an input data corresponds to an image patch, whose semantic label is associated with its center pixel.

⁷Each convolution block consists of multiple functional layers including a convolutional layer (**conv**), a ReLU layer, optionally a pooling layer (**pool**) or a batch normalization layer (**bn**).

feature representation (from convolutional blocks) and generates a class likelihood vector for the task at hand. Different from previous methods that usually adopt hand-crafted features as well as a separate classifier for image analysis [66][67][68][69], CNN is a very elegant paradigm for image analysis that extracts representative image features as well as learns discriminative classifier in a joint framework. Most importantly, with the great advancement of computational facilities (i.e. Graphical Processing Unit - GPU), CNN can be efficiently trained and inferred on modern GPUs. The seminal success of CNN in large-scale object recognition challenge spurred its resurgence in computer vision. It's therefore not surprising that recent successes of CNNs have spanned to wide variety of image analysis problems including image classification [23][14][24][13][19][70], object detection [71][72][73][74][75], semantic segmentation [76][21][77][22][34], human pose estimation [78][79][80][81] and etc. Specifically in scene parsing, Farabet et al. [47] pioneered employing CNN to performance scene paring.⁸ In detail, their proposed convNet takes input as multi-scale raw image patches, and it is trained to minimize the average cross entropy over image pixels (patches.) Ultimately, they presented very interesting results on real-world image scene labeling benchmarks, which significantly outperforms highly-engineered low-level feature based segmentation system. Furthermore, Pinheiro et al. [56] trained a single-scale CNN based on small-size image patches, and apply this CNN to large-size patches recurrently so that more context is engaged in semantic prediction. Recurrent CNN also presented very promising segmentation results on scene parsing benchmarks. Mostajabi et al. [82][48] extracted features from different scopes of image regions, and then concatenated them to yield the context-aware representation. Therefore, regional and global context is encoded in the local representation. On public benchmarks, all these CNN-based segmentation models demonstrate performance superiority over Random Forest-based and hand-crafted feature-based segmentation models.

⁸As each image patch is associated with a semantic class label, scene labelling can be simply regarded as a patch analysis task (c.f. image analysis in object recognition or image classification), if no correlations among image patches are explicitly considered in such analysis.

Fully Convolutional Network (FCN) is a special form of convolutional neural network that does not have fully connected layers. It is firstly proposed by Long et al. [21], and ever since then it has been broadly used in dense image analysis. Here in this chapter, FCN is specifically referred to convolutional networks adapted from pre-trained CNN. In practice, FCN achieves breakthrough performance that dramatically outperforms previous patch-CNN (trained from scratch) as well as previous segmentation systems that relies on hand-crafted features. The noticable performance superiority of FCN should be attributed to the robust high-level feature transfer from pre-trained CNN. Not surprisingly, All state-of-the-art segmentation systems are FCN based, such as FCN [21] DeepLab [42] [22] [83], DilatedNet [34] [84], RefineNet [35], PSPNet [46], etc.

Generally, CNN / FCN demonstrate significantly more powerful learning capability to learn discriminative and representative features for image patches. Thus, we adopt this machine learning paradigm throughout this thesis.

2.2.2 Probabilistic Graphical Models

Probabilistic graphical models are able to enforce label consistency over local regions. It is an very interesting and important research direction in computer vision and machine learning community. In general, image units (pixel or superpixel) are densely connected in a local region (clique), and then a label compatibility function (potential) is to evaluate how semantic labels (within the clique) agrees with some manually-defined priors / rules⁹. An overall energy function (potential) sums over all potentials of local cliques. Therefore, minimizing the energy is to implicitly enforce that all pixels should take a label that is compatible to the rest image pixels. In scene segmentation, many researchers successfully adopted graphical models to enhance semantic consistency of output label maps. For example, Shotton et al. [85] formulated the unary and pairwise features in a 2nd-order

⁹A common used prior is that neighbourhood pixels are supposed to take the same semantic labels unless their appearance differs dramatically. Although not so accurate, this prior always leads to significantly more visually pleasing segmentation maps.

sparse Conditional Random Fields (CRF) graphical model. Roy et al. [86], Zhang et al. [87] built a fully connected pairwise CRF to enforce higher order labeling coherence. Krahenbuhl et al. [41] significantly speeded up the inference of fully connected CRF by using Gaussian potentials as well as a highly efficient convolution data structure – permutohedral lattice. Kohli et al. [88], Kotschieder et al. [62] and Marquez et al. [89] modeled the higher order pixel correlations by considering patch/superpixel as a clique. He et al. [90] defined a multi-scale CRF that captures different contextual relationships ranging from local to global granularity. Recently, Chen et al. [42] and Zheng et al. [43] applied fully connected CRF [41] to pixel-wise class likelihood maps, and they generate significantly smoother label regions as well as sharper boundaries in the final label prediction maps on Pascal VOC dataset.

In general, using graphical models is a promising way to enforce label consistency ranging from different levels. Unfortunately, higher-order (high-level) label consistency involves very dense graphical structure, and minimizing the resulting labelling energy (potential) is computationally demanding or even impossible in some situations. In practice, many graphical models simply model pairwise potentials, thus their model capacities degenerate to enforce neighbourhood coherence. Nowadays, fully connected CRF has been considered to be the *de facto* post-processing method to improve the segmentation accuracies. Although effective, they are rarely considered to be real-time applicable as the inference of graphical model is extremely time-consuming.

From the perspective of enforcing label consistency, our work is related to this branch of works. First, we explore using global context to encourage high-level label consistency, and this global scene prior can be efficiently estimated in a non-parametric label transfer model. Then, we formulate two computational layers – DAG-RNN and Convolutional Context Network (CNN) – to aggregate high-level context to enhance feature representation, thus high-level label consistency is also achieved implicitly. In terms of efficiency, our proposed computational layers are significantly faster. Besides, our proposed methods can model higher-order dependencies which are difficult to be captured

in current graphical models. Meanwhile, our segmentation models can also be benefited from applying fully connected CRF [41] as post-processing module, which demonstrates the orthogonality of our research endeavour with this line of works.

2.2.3 Label Transfer Models

Non-parametric label transfer model [1, 2, 91, 92, 93, 94] used to be a very popular and effective scene labelling pipeline due to its outstanding performance (in terms of speed and segmentation accuracy) and the scalability to large scale data ¹⁰. They estimate the class likelihood of image units (e.g. pixels, superpixels or image patches) from the statistics of image units over several globally similar images. Technically, global scene features are firstly utilized to retrieve a set of images which are expected to share very similar scene layout with the testing image. Next, statistics over their corresponding label maps are leveraged to estimate the class likelihood of image units. In this regard, the output label maps generally exhibit somewhat accurate scene layout. The pioneering label transfer work [1] transformed RGB images to SIFT [51] images, which were then used to seek correspondences over pixels. Next, an energy function was defined over pixel-level correspondences, and the label prediction maps are obtained by minimizing this energy. The SuperParsing system [2], on the other hand, presented a very simple and effective non-parametric model to achieve robust label transfer. It estimated the class probability of testing image superpixels by transferring simple statistics over image superpixels from retrieved image sets. Further, Eigen et al. [91] improved this framework by learning adaptive weights for each low-level features, and it resulted in better nearest neighbor search. Singh et al. [92] augmented the global feature with semantic context features estimated from first-round labeling result, which also led to higher-quality retrieved image sets. Meanwhile, Yang et al. [94] achieved impressive result on average class accuracy

¹⁰Due to the non-parametric characteristics of those models, they are not required to be re-trained whenever extra training data is added.

by augmenting the image-unit level nearest neighbour retrieval with more rare-class instances. Gould et al. [95, 96] built a graph for dense image patch and superpixel to achieve effective label transfer.

We adopt this framework mainly to achieve the global context incorporation into patch CNN. In general, label transfer models are able to roughly estimate the correct scene layout, although many small-size objects are usually ignored. This observation largely motivates us leveraging the strengths of non-parametric label transfer models to regularize the patch CNN-based local labelling system which is relatively weak in terms of interpreting scene layouts. Meanwhile, in comparison with these works that are based on hand-crafted features, we used the learned CNN features which are very compact and discriminative. We expect our learned features to benefit their systems in terms of accuracy and efficiency as well.

2.2.4 Model Ensemble

The ensemble methods [97][98] have achieved great success in machine learning. The idea is to build a strong predictors by assembling many weak predictors. The assumption is that these weak predictors are complementary to each other when they are trained with different subset of features and training data. Some examples are random forest [60][59], bagging [99], boosting [100], etc. Random forest has been successfully used in solving image labeling problems. For example, Shotton et al. [98] learned an ensemble of decision jungles to output the semantic class labels of pixels. Kotschieder et al. [62] constructed a random forest that directly maps the image patches to their corresponding structured output maps. Meanwhile, model ensemble [23][13][24] has been a common bells and whistles to further boost the performance. It has been pervasively adopted in the large-scale ImageNet competition [14] as well as COCO challenge [44]. Specifically, many deep neural networks (their network architecture is identical) ¹¹ are trained, and the late fusion

¹¹For example, GoogLeNet [24], VGG-16 [13], ResNet [19], Inception-ResNet [25], ResNext [26], DenseNet [27], Xception [28] for image / scene classification; Faster R-CNN [73], Mask R-CNN [101],

of their decisions give rise to the final output. By doing this, the ensemble model is able to enhance its classification (or localization, detection, segmentation, etc.) accuracy significantly.

In this thesis, we explore using model ensemble to address the severe data imbalance in scene segmentation. Specifically, we train multiple CNN with different patch sampling strategies, and then we fuse predictions over these different and complementary CNN components. As a whole, we name this set of CNN as CNN-Ensemble. Our CNN-Ensemble is different from simple model ensemble methods. Concretely, the data that are used for training each individual model are either exactly the same or sampled without discrimination in previous cases. Whereas in CNN-Ensemble, data are sampled differently (data statistics is totally different in terms of frequency distribution) for training each CNN component, which leads to significantly different behaviour of each CNN at predicting accuracy for rare and frequent classes.¹²

2.2.5 Contextual Modeling

Contextual modeling is of great significance to contextualize features, and it usually leads to dramatic enhancement of high-level representative capability of local features. Therefore, contextual modeling has been broadly investigated and adopted to phenomenally improve the semantic segmentation performance. Generally, there are two orthogonal research directions that achieve this functionality.

One branch of work introduces new computational layers to achieve contextual modelling (especially in the deep learning era), which are usually placed on top of a pre-trained CNN (feature extractor) to construct an end-to-end trainable segmentation network. For example, Liu et al. [29] adopted local convolution layers to approximate the mean field algorithm for pairwise terms in deep parsing network (DPN). Lin et al. [30] added

SSD [75] [102], YOLO [74] [103] for object detection; FCN [21] DeepLab [42] [22] [83], RefineNet [35], PSPNet [46] for image semantic segmentation, etc.

¹²In this regard, CNN-Ensemble integrates strengthens of all CNN components, and ultimately performs competitively well for predicting both frequent as well as rare classes.

pairwise connections between local features and employed CNNs to model the semantic compatibility between image regions, and they achieved state-of-the-art results on object segmentation dataset. Visin et al. [31] employed 1-D recurrent neural networks (RNNs) to propagate local context in feature maps. Chen et al. [22] proposed an atrous spatial pyramid pooling (ASPP) network to aggregate multi-scale context for feature maps. Recently, Yu et al. [34] introduced a convolutional network using dilated convolution kernels (atrous convolution [42]) to perform context aggregation over class likelihood maps. Sharma et al. [32] used recursive neural networks to aggregate global context and then disseminate it to local regions. Liu et al. [33] proposes ParseNet to aggregate global context via global pooling layer. All these computational layers are designed to encode extra context into local features, and they do bring in significant performance improvement in practice.

Another representative branch of work is to leverage probabilistic graphical models such as fully connected CRF [41] (c.f. Section 2.2.2) to contextualize the unary predictions of segmentation networks. For example, Chen et al. [22] applied CRF to the unary predictions of DeepLab network, and they observed obvious improved visual quality of parsing maps. Subsequently, Zheng et al. [43] formulated CRF as a Recurrent Neural Network (CRF-RNN) so that it can be jointly trained with segmentation networks.

Our work in this thesis is more related to the first batch for works. To be more specific, we propose two different computational layers that perform high-level context aggregation: DAG-RNN and convolutional context network (CCN). We demonstrate that these proposed layers represent two typical state-of-the-art context aggregation module. As an illustration, we also experimentally demonstrate in more detail how DAG-RNN differs from CRF in terms of context aggregation in Section 4.8.3. It's however important to mention that those two line of research works are orthogonal towards achieving context aggregation, and combining these two components should give improved results. In our experiments, we do observe that fully connected CRF consistently improves the performance our segmentation networks (e.g. DAG-RNN).

2.2.6 Recurrent Neural Networks

One branch of works interpret recurrent model in an iterative manner. In detail, they generally refer to applying the same model recurrently during different iterations. For example, Pinheiro et al. [56] attached the RGB raw data with the output of the Convolutional Neural Network (CNN) to produce the input for the same CNN in the next iteration. Li et al. [104] and Tu et al.[105] augmented the patch feature with the output of the classifier to be the input for the next iteration, and the classifier parameters are shared across different iterations. In general, they model the context in the form of intermediate outputs (usually local beliefs), which implicitly encodes the neighborhood information.

In this thesis, recurrent neural networks (RNNs) refer to the class of networks that have ‘memory’, which largely differentiate the above branch of recurrent models [56] [105]. Besides their success in modeling temporal dependencies in speeches [17] and languages [106] [107] [108] [109], RNNs have recently been adopted in computer vision community. Here we only review literatures that adopt simple RNNs in 2-D images. Visin et al. [110] adopted ReNet, which is a stacked of 1D-RNN to perform image classification. Zuo et al. [111] placed 1D-RNN on top of pre-trained CNN to model weak contextual dependencies over image regions for image classification, and Shuai et al.[112] also adopted 2D-RNN [16] to model dependencies over image features. Both of them observe promising performance gains after incorporating RNNs. Recently, Tai et al.[113] and Zhu et al.[114] demonstrated that considering rich structure (i.e. tree) is beneficial for modeling the global representation of sentences. Our proposed DAG-RNN is a generalization of chain-RNNs [115][106], recent tree-RNNs [113][114] and 2D-RNNs [16][112], and it enables the network to model long-range semantic dependencies for graphical structured images.

In this thesis, Chapter 4 also explores generalizing RNNs to model rich contextual dependencies over image regions. The most relevant work to ours is ReSeg [116], which

stacks a ReNet [110] on top of a pre-trained CNN to perform semantic segmentation. ReNet is also used to capture the rich dependencies over image features. Different from ReNet that uses 1D-RNN to propagate local context horizontally or vertically, our DAG-RNN enables RNN work for 2D images. This has significant advantages in terms of modeling rich spatial dependencies over 2D image regions. Our DAG-RNN can naturally model the 2D contextual dependencies, whereas 1D-RNN can only model horizontal or vertical contextual dependencies. Meanwhile, considering that ReSeg only use first conv block of pre-trained CNN, ReNet needs to process significantly denser feature maps than DAG-RNN, which makes the inference (and training) of ReSeg dramatically less efficient. To directly compare with ReSeg [116], we apply DAG-RNN on CamVid [85] which is a small-scale urban road scene parsing dataset. DAG-RNN achieves 63.4 % mIOU versus 58.8% mIOU (reported by ReSeg) ¹³.

2.2.7 Retaining / Recovering Detailed Information

As mentioned earlier, state-of-the-art segmentation networks [21][46][45][22] are adapted from pre-trained classification networks (e.g. CNN trained from ImageNet [14]). Due to subsampling layers (e.g. pooling, convolutions with stride larger than 1) in pre-trained CNNs, detailed low-level image cues are gradually lost in higher-level feature maps. However, these low-level information is essential to detect small-size objects as well as to accurately localize region boundaries. Thus, it's important to retain these precious lower-level image information in high-level feature maps. The seminal work - FCN [21] - demonstrated that more detailed parsing maps can be generated as lower-level feature maps were explicitly incorporated into predictions. From the perspective of architecture designs, this is achieved by adding skip layers ¹⁴ from lower-level feature maps of pre-trained CNN. Different from this practice, DeepLab [22] and DilatedNet [34]

¹³It is worth mentioning that DAG-RNN is trained with a new proposed loss, which is expected to slightly improve the overall segmentation accuracies.

¹⁴From the functionality perspective, skip layers can be interpreted as a classifier.

[84] advocated to mitigate the loss of detailed spatial information by removing some of downsampling layers in pre-trained CNN. Afterwards, they performed the subsequent convolution operations with higher dilation factors (i.e. stride rate) so that parameters in pre-trained CNN can be still reusable. Most recent segmentation networks are based on the latter architecture design due to its superior parsing performance, such as [40][22][45][46].

In addition, researchers have also been dedicated to developing networks (i.e. decoder network) [21] [37][38] [39] [40] to recover the lost spatial image details. Functionally speaking, these decoder networks are used to upsample the coarse feature maps (i.e. class likelihood maps) so that high-resolution prediction maps can be generated by segmentation networks. In terms of architecture design, they are usually placed at the end of the segmentation networks (c.f. Figure 2.1.). For this purpose, researchers have carefully designed multiple different architectures for decoder networks. For example, Long et al. [21], Badrinarayanan et al. [37] stacked multiple learnable deconvolution (convolution transpose) layers to upsample the final score maps. Noh et al. [38], on the other hand, constructed a deep deconvolution network that is symmetric to pre-trained CNN by using un-pooling layers and deconvolution layers. They demonstrate that the proposed deconvolution network is able to largely recover image details. Different from leveraging deconvolution layers to build the decoder network, Wang et al. [40] proposed to use convolution layers (with high-dimensional channels) to achieve this goal. As claimed by the authors, the high-dimensional features gradually recover the spatial information.

It’s important to note that these two line of researches are orthogonal to preserve detailed image information. In this thesis, our main focus is how to effectively incorporate lower-level information into local labelling. As will be demonstrated in Chapter 5, we investigate the limitations of current parametrization of skip layers. Meanwhile, we propose “dense skip” architecture to adapt pre-trained CNN so that rich set of lower-level image information is explicitly engaged in dense semantic prediction. Meanwhile, we

simply use learnable deconvolution networks as in FCN [21] and SegNet [37] to perform feature map upsampling.¹⁵

2.2.8 Multi-scale Aggregation

In order to address the huge scale / size variances of objects in computer vision, multi-scale aggregation either in feature level or in class-likelihood level are common practices in computer vision tasks. Technically, there are two well-known approaches to achieve this goal. To elaborate,

- **Image pyramid approach [47][30]:** First, multi-resolution image copies (correspond to an image pyramid) are generated. Then, each image copy is sent to the model (e.g. segmentation networks) for processing. Finally, their output can either be concatenated (i.e. feature level) or be element-wise fused together (classification score level).
- **HyperColumn approach [117][118]:** It is specifically applicable to CNN based segmentation networks. Concretely, one image is passed to the model for processing, and then outputs (i.e feature maps) from different intermediate layers are concatenated and they are together fed into a classifier for evaluation.

Both approaches are effective in terms of aggregating multi-scale image cues, and they can also be used to build end-to-end trainable segmentation networks. Unfortunately, both approaches are expected to incur either significantly higher computation time or larger memory consumption¹⁶. Hence, very limited scale contexts are actually aggregated in practice, which largely degrades the model capacity of modelling object scale invariance.

¹⁵This is a relatively efficient architecture for decoder network, and it is easy to be implemented. It's important to note that the non-learnable deconvolution networks (with bilinear kernel initialization) performs equally well with learnable deconvolution networks.

¹⁶In image pyramid scenario, the inference / training time can be increased to N folds when there are N multi-resolution image copies. Meanwhile, in hypercolumn case, the concatenation operation (over features) is expected to result in very high-dimensional features, which greatly increases the memory overhead.

For example, Farabet et al. [47] and Lin et al. [30] simply build a 3-layer image pyramid. The PixelNet [117] simply concatenates 5-scale intermediate features for dense semantic prediction.

In this thesis, as will be elaborated in Chapter 5, we incorporate very rich scale of feature maps into dense semantic labelling. For example, in VGG-16 based segmentation network, 13-scale feature maps are engaged in semantic predictions, whereas PixelNet [117] only includes 5-scale feature maps. Most importantly, we adopt element-wise fusion over classification scores, which marginally increases computational overhead as well as memory consumption. Concurrent research works [75] [36] [119] in object detection community adopts a similar approach to aggregate multi-scale feature maps.

2.3 Summary

In a nutshell, this thesis discusses major research directions in scene segmentation in the following aspects.

- We differentiate scene segmentation from object segmentation tasks, and we figure out three key challenges that are inherent to scene segmentation. In align with these challenges, we develop three segmentation models, each of which addresses part of these challenges.
- We extensively discuss how to effectively incorporate informative context into deep neural networks. We have discussed, compared our context aggregation layers with other state-of-the-art works technically and experimentally.
- We explore using ensemble methods to address data imbalance, which has rarely been discussed by previous works. Besides, our ensemble method differs from previous literatures in terms of motivation as well as technical details.

- We also discuss different ways to preserve detailed spatial information in FCN based segmentation networks. We examine and compare current methods to retain detailed information, based on which, we propose a significantly more effective and efficient architecture to improve low-level semantic parsing performance.
- We extensively compare our segmentation networks with state-of-the-art networks quantitatively (speed and accuracy) as well as qualitatively.

Starting from next chapter, we are going to elaborate the technical details of the three proposed segmentation models individually.

Chapter 3

Patch-based convolutional networks

3.1 Introduction

The resurgence of Convolutional Neural Networks (CNNs) [23, 120] greatly revolutionizes the computer vision community due to their breakthrough performance in a wide variety of high-level image analysis tasks [23, 121, 122, 123, 78, 124] . More specifically in the community of scene segmentation, Farabet et al. [47] and Pinheiro et al. [56] pioneered to adopt CNNs to scene labeling. In this scenario, CNNs are used to model the class likelihood of pixels directly from local image patches, and we call them patch-CNNs. They are able to learn strong features and classifiers to discriminate the local visual subtleties. In comparison with methods that adopt hand-crafted features and separate classifiers, patch-CNNs can produce satisfactory labeling results in a more elegant and efficient framework.

In general, single CNN ¹ fails to produce satisfactory labeling results due to the severely imbalanced class frequency distribution in natural scene images. To address this issue, we propose **CNN-Ensemble**, which aggregates predictions from different and complementary CNNs. Each CNN component shares the identical network architecture, but it is trained from image patches with disparate statistics, which are generated from different sampling methods. In the end, the proposed CNN-Ensemble is capable of yielding more reliable labeling results than any single CNN.

¹Hereafter in this chapter, CNN simply refers to patch-CNN.

Figure 3.1: Motivation of our integration model: the parametric model can distinguish visually different pixels very well, but get confused for pixels that are visually similar in local context. However, the local ambiguities can be easily eliminated as long as the correct global scene semantics is revealed. A more consistent labeling result can be achieved by integrating their beliefs. The figure is best viewed in color.

Even though powerful, CNN-Ensemble still struggles in differentiating visually similar pixels as it only consider limited context in local classification. As exemplified in Figure 3.1, the sand pixels are confused with road and sidewalk pixels in a local view. We refer to such problem as *local ambiguity*. This problem has usually been addressed from two perspectives:

- Augmenting the scale of context to represent pixels: [47] considers multi-scale context input, [56] increases the size of context input in a recurrent CNN framework. These methods somehow mitigate the local ambiguity, however they may have a negative effect to small-size objects and may also degrade the efficiency of the system.

- Building a probabilistic graphical model to capture the explicit label dependencies among pixels [90, 88, 85, 87]. However, the parametric graphical model is usually hard and inefficient to optimize when the higher order potentials are involved, and the low-order potentials suffer from low representation power.

Here in this chapter, we propose to utilize the global scene semantics to eliminate the *local ambiguity*. As a simple example in Figure 3.1, the confusion between ‘road’ and ‘sand’ pixels can be easily removed if the global “coast” scene is revealed. Intuitively, a global scene constraint is implicitly enforced to allow more reliable local classification. Such global constraint is mathematically achieved by adding a global energy term to the labeling energy function. However, due to the extraordinarily huge labeling space, it’s infeasible to model the global energy parametrically. Thus, the global energy is practically modeled in a non-parametric framework by transferring the class dependencies and priors from its global similar exemplar images.

Furthermore, a large margin based metric learning objective is introduced to fine tune the network, thus making the estimation of global belief more accurate. Finally, the class likelihood of pixels are obtained by integrating the local and global beliefs. Based on which, our integration model outputs the label prediction maps. We justify our method on the popular and challenging scene parsing benchmarks: SiftFlow [1] and Barcelona [2] datasets. Even without any post-processing, our integration model is able to achieve very competitive results that are on par with the state-of-the-arts. Overall, the contributions of this chapter are summarized as follows:

- (i) We propose the **CNN-Ensemble**, in which each CNN component concentrates on learning distinct visual patterns. The aggregation of single CNNs gives rise to much more powerful model that is able to generate more reliable labeling maps.
- (ii) We leverage global scene semantics to remove the *local ambiguity* by transferring class dependencies and priors from similar exemplars.

- (iii) We introduce the CNN metric, and show that the learned metrics are beneficial in our non-parametric global belief estimation.

The rest of this chapter is organized as follows: Section 3.2 presents the formulation of our integration model are presented; Then, details of estimating the local and global beliefs are elaborated in Section 3.3 and 3.4 respectively; Section 3.5 demonstrates the experimental setup and reports the results of the proposed methods; Section 3.6 concludes this chapter.

3.1.1 Ensemble Models

The ensemble methods [97][98] have achieved great success in machine learning. The idea is to build a strong predictors by assembling many weak predictors. The assumption is that these weak predictors are complementary to each other when they are trained with different subset of features and training data. Some examples are random forest [60][59], bagging [99], boosting [100], etc. Random forest has been successfully used in solving image labeling problems. For example, Shotton et al. [98] learned an ensemble of decision jungles to output the semantic class labels of pixels. Kotschieder et al. [62] constructed a random forest that directly maps the image patches to their corresponding structured output maps. Our CNN-Ensemble is different from these works. First, the individual model in theirs are very weak, whereas each of our single CNN has very strong capability. Second, the data that are used for each individual model training in their works are sampled without discrimination. In contrast, data that are fed to each single CNN are sampled differently, therefore their statistics are different.

Recently, the ensemble models [23][13][24] have been pervasively adopted in the large-scale ImageNet classification competition [14]. Specifically, many deep neural networks (their network architecture is identical) are trained, and the fusion of their decisions give rise to the output. By doing this, the ensemble model is able to enhance its classification accuracy slightly. Our CNN-Ensemble also differs from these works. In their works, each

network component is trained with exactly the same data, and the difference of network components mainly originates from nonidentical network initializations. In contrast, every CNN component is trained from entirely different data, which will guide the CNN component to focus on learning different but complementary visual patterns.

3.2 Formulation

The image labeling task is usually formulated as a discrete energy minimization problem. Specifically in this chapter, we consider minimizing the following energy:

$$E(X, Y) = E_I(X, Y) + E_G(X, Y) \quad (3.1)$$

where $X = \{X_1, X_2, \dots, X_N\}$ is the observed image and X_i corresponds to the i th pixel; $Y = \{Y_1, Y_2, \dots, Y_N\}$, $Y \in \{1, 2, \dots, L\}^N$ denotes a labeling configuration for image X and L represents the cardinality of classes; $E_I(X, Y)$ and $E_G(X, Y)$ are the local and global energy term respectively.

Individually speaking, the local energy term $E_I(X, Y)$ measures the likelihood of image X taking the labeling configuration Y in the local view. Mathematically, it is expressed as the summation of local unary potential $\Psi_I(X_i, Y_j)$:

$$E_I(X, Y) = \sum_{X_i \in X} \Psi_I(X_i, Y_j) \quad (3.2)$$

where $\Psi_I(X_i, Y_j) = -\log(P_I(X_i, Y_j))$ is defined as the negative log likelihood of pixel X_i being labeled as Y_j ; Hereafter, we call $P_I(X_i, Y_j)$ the local belief.

The global energy $E_G(X, Y)$, on the other hand, reflects the likelihood of image X taking the labeling configuration Y in the global view. A naive implementation is to consider all the pixels to be in a single clique, and $E_I(X, Y)$ evaluates the energy term according to the labeling states. However in this scenario, the labeling states are prohibitively huge (L^N), it makes the energy evaluation intractable in practice. Inspired

Figure 3.2: Framework of our integration model: the parametric CNNs are responsible for emitting the local beliefs, and the non-parametric model is practically used to output the global beliefs. The integration of local and global beliefs gives rise to the un-normalized class likelihood of pixels. The modules painted in yellow represents parametric models (CNN and Metrics).

by [89], we adopt an non-parametric approach to decompose the global energy to the aggregation of global unary potential $\Psi_G(X_i, Y_j)$:

$$E_G(X, Y) = \sum_{X_i \in X} \Psi_G(X_i, Y_j) \quad (3.3)$$

where $\Psi_G(X_i, Y_j) = -\log(P_G^{\mathcal{S}(X)}(X_i, Y_j))$; $P_G^{\mathcal{S}(X)}(X_i, Y_j)$ denotes the likelihood of X_i being Y_j in the global scene view, and it is estimated from images in $\mathcal{S}(X)$ which are expected to share similar global scene semantics with image X . Likewise, we call $P_G^{\mathcal{S}(X)}(X_i, Y_j)$ the global belief hereafter. As shown, the global energy implicitly captures the pixel dependencies via transferring scene semantics from images in $\mathcal{S}(X)$. As an example in Figure 3.1, the global scene exemplars ($\mathcal{S}(X)$) define a "coast" scene, in which road and sidewalk pixels are invalid and sand pixels are more likely to appear in the bottom regions. Finally, by rewriting the energy functions, we generate the following form:

$$E(X, Y) = - \sum_{X_i \in X} \log((P_I(X_i, Y_j) \cdot P_G^{\mathcal{S}(X)}(X_i, Y_j))) \quad (3.4)$$

The above energy is numerically proportional to the integration of beliefs from two sources: (1), Local belief: $P_I(X_i, Y_j)$ measures the belief for pixel X_i based on its surrounding local context; (2), Global belief: $P_G^{\mathcal{S}(X)}(X_i, Y_j)$ denotes the belief for X_i from the global scene view. An intuitive interpretation of Equation 3.4 is that a global scene constraint (prior) is enforced to the local classification in the form of weighting the local beliefs of pixels with their corresponding global beliefs. Since the estimation of class likelihood for different pixels is independent, the energy minimization (inference) can be done in an efficient pixel-wise manner: $Y = \bigcup_{i=1:N} Y_i$, $Y_i = \operatorname{argmin}_{1,\dots,L} E(X_i, Y_j)$.

The pipeline of our model is depicted in Figure 3.2. Systematically, an image is first passed to the truncated CNN, and the corresponding pixel feature maps are generated. Next, the feature maps are fed into two branches: (1), they are independently classified based on the parametric CNNs (CNN-Ensemble), which yield the local beliefs; (2), they

are aggregated to produce the global scene envelop, which is used to retrieve the global similar exemplar images. Based on which, the global belief is estimated. Finally, the integration of local and global beliefs yields the un-normalized class likelihood of pixels, based upon which the integration model outputs the label prediction map. We elaborate each module in the following sections.

3.3 Local Beliefs

In a local view, the semantic class of each pixel X_i is determined by its surrounding image region (patch). A parametric model is commonly used to emit its local belief $P_I(X_i, Y_j)$. in this chapter, this model is parameterized by **CNN-Ensemble** - an ensemble of Convolutional Neural Network (CNNs). The CNN components are trained from entirely different image regions (patches) in terms of the class frequency distribution, thus they capture complementary image statistics. In detail, each CNN component is enforced to focus on discriminating some specific classes by adaptively learning different features and classifiers. By fusing these complementary cues, CNN-Ensemble is expected to output more reliable labeling results.

3.3.1 Parametric CNNs

The Convolutional Neural Networks (CNNs) learn features and classifiers in an end-to-end trainable system. They are able to learn compact yet discriminative representations, which are easy to be differentiated for the jointly learned classifiers. Specifically in the image labeling task, the parameters of CNNs are optimized based on the image patches, whose labels are associated with the centering pixels. Unlike other CNNs [47] [56] which are fed with multi-scale or large field-of-view patches, we use moderate-size contextual windows to predict the labels for pixels. By doing this, we enforce the network to learn strong representations to discriminate the local visual subtleties. Meanwhile, the locality information is well preserved, which is crucial to differentiate small-size object classes.

Moreover, as evidenced by the experiments later, our CNNs outperform their nets [47][56] dramatically. In addition, our CNNs are more efficient in terms of inference.

The architecture of our CNNs is demonstrated in Figure 3.2. It accepts 65×65 image patches as valid input. If we assume that the last fully connected layer (FC-2) serves as the functionality of classifiers, the removal of which in the CNN gives rise to the Truncated Convolutional Neural Network. In other word, if we pass an image patch (65×65) to the truncated CNN, its output is a representation vector (64 dimension), which summarizes the contextual information of the input patch. In this perspective, truncated CNN can be interpreted as a feature extractor.

3.3.2 Data Sampling

Knowing that the number of training patches are prohibitively huge (thousands of millions), we only use a fraction of them for the network training. Specifically, at the beginning of each epoch, training patches are randomly sampled. By doing this, we are able to decrease the training time dramatically. Meanwhile, the sampling strategy does not significantly harm the performance, as the image patches are highly redundant (patches that are related to neighborhood pixels are usually the same) and the randomness injected into the data sampling during each epoch enables the network to “see” the whole data throughout the whole training process. Experimentally, we have tried using different number sampling patches (from 10k to 50k), and we don’t observe significant performance disparity. Here in this chapter, we introduce four sampling methods based on the class frequency distribution in natural scene images:

- **Global sampling (GS):** It samples patches randomly from the collection of all the training patches. Then the class frequency distribution in the sampled patches should be very close to that in the original images.

Figure 3.3: The first graphic shows an example of the class frequency distribution in natural scene images, through which we can observe that large percentage of pixels belong to very small number of frequent classes. The second figure delineates the class-wise accuracy curves for CNN-M. The experimental statistics are based on the SiftFlow dataset [1].

- **Class sampling (CS)**: It samples patches in a manner that classes appear equally in the resulting sampled patches. Note that some patches may appear multiple times if they are from extremely rare classes.
- **Hybrid sampling (HS)**: It is a compromise between global sampling and class sampling. In detail, it firstly samples patches globally, and then augments the rare-class patches until their occurrence frequencies reach the desired threshold η in the sampled patches.
- **Truncated class sampling (TCS)**: It adopts the same sampling procedure as class sampling, but removes all the frequent-class patches.

The threshold η is used to determine whether a class is frequent or rare: the class is considered to be frequent as long as its occurrence frequency in the training data is above η , otherwise, it belongs to a rare class. Obviously, the above sampling methods are expected to yield significantly different sampled patches in the form of class frequency distribution. When these image patches are presented in the network training phase, the CNNs are enforced to learn disparate visual patterns, thus they behave differently. We next discuss how the sampled image patches influence the characteristics of CNNs.

3.3.3 CNN-Ensemble

CNNs, even with the identical architecture, can be functionally different if they are trained from image patches with disparate data distribution. Concretely, to fit the image patches whose statistics are not the same, CNNs adaptively learn different representations and classifiers. In consequence, it leads to the big performance discrepancy on labeling the same images. Let's denote **CNN-M** the CNN trained with image patches sampled through method M. As shown in Figure 3.3, CNN-M exhibits significantly different characteristics towards predicting the semantic classes of pixels: **CNN-GS** prioritizes

the frequent classes, therefore it optimizes the overall pixel accuracy; **CNN-CS** implicitly normalizes the weights of classes by performing downsampling and oversampling operations to frequent and rare classes respectively, thus it maximizes the average class accuracy; In comparison with CNN-GS, **CNN-HS** gives slightly higher weights to rare classes, hence it compromises the two above criterions; **CNN-TCS** works exceptionally well on differentiating rare classes.

To produce a satisfactory labeling prediction map, the algorithm is required to perform extraordinarily well to correctly predict frequent classes, and in the mean time should work well towards recognizing rare classes precisely. The former criterion guarantees that the scene semantics of the images are well defined, and the latter one enforces that objects (rare classes) are not missing in the scene. However as manifested in Figure 3.3, none of the CNN-M satisfies both criterions. Therefore, we propose the **CNN-Ensemble**, which combines the predictions from the CNN-M components. Mathematically, the local belief $P_I(X_i, Y_j)$ is derived with the following equation:

$$P_I(X_i, Y_j) = \frac{1}{M} \sum_{m=1}^M P_m(X_i, Y_j, \Theta_m) \quad (3.5)$$

where M is the number of CNN-M components, $P_m(X_i, Y_j, \Theta_m)$ denotes the class likelihood prediction from the m -th CNN-M and Θ_m represents the learnable parameters of the corresponding CNN. As mentioned, CNN-M components are designed to focus on distinguishing some specific classes. For example, CNN-GS discriminates frequent classes excellently, and CNN-TCS captures the subtleties of infrequent classes. The proposed CNN-Ensemble fuses the complementary predictions from different sources. Our later experiments in Section 3.5 will demonstrate that CNN-Ensemble is able to produce much more reliable labeling results quantitatively, and it performs excellently on both overall pixel accuracy and average per-class accuracy.

3.4 Global Beliefs

In a global view, the semantic class of each pixel X_i in image X is determined by the global scene semantics of X . In other words, its global class likelihood $P_G(X_i, Y_j)$ should match the expected scene layout of X . Even though the pixel X_i is represented identically, its class belief could vary significantly depending on how the pixel is evaluated. Take the simple example as an illustration in Figure 3.1, the pixels in the lower part of the image could be ‘sand’, ‘road’ or even ‘rock’ in a local view. We refer to this problem as *local ambiguity*. In contrast, with the awareness of global ‘coast’ scene, it is obvious that ‘sand’ class is preferred in a global view. In this perspective, the global scene prior is a good remedy to alleviate the local ambiguity.

3.4.1 Non-parametric Global Belief Transfer

As elaborated, the parametric CNNs (CNN-Ensemble) are able to produce good labeling results for the pixels with good local contextual support, they still suffer from the notorious *local ambiguity* problem. Previously, researchers usually addressed this issue by generating contextual aware local features. For example, Farabet et al. [47] fed the network with multi-scale image patches to yield richer contextual aware local features, and likewise Pinheiro et al. [56] took the network input as larger image patches. In this chapter, the local disambiguation is achieved by enforcing a global scene constraint to local classification. More specifically, a pixel is considered under global context: the class likelihood of pixels should satisfy the scene layout and semantics of the image.

First, we generate the global-level representation for the considered image $X \in \mathbb{R}^{h \times w \times 3}$, where h, w is the height and width of the image X . The corresponding CNN feature tensor $F \in \mathbb{R}^{h \times w \times M}$ can be obtained by passing densely sampled image patches in X to the truncated CNN ($M = 64$ in our implementation). Next, we introduce the average pooling operator *pool* [23] to aggregate the pixel features, thus giving rise to the global feature H . In detail, suppose an image is decomposed to regions $\mathcal{R} = \{R^{(1)}, R^{(2)}, \dots, R^{(J)}\}$

², the region feature is generated by applying *pool* operator to the constituent pixel features: $H(R^{(i)}) = \text{pool}(F^i), \forall i \in R^{(i)}$. The global image representation is defined as the concatenation of region features $H = [H(R^{(1)}), H(R^{(2)}), \dots, H(R^{(J)})]$. As expected, this global image feature H not only conveys discriminative scene semantics but also encodes scene layout information.

Then, based on H , the global nearest exemplars $\mathcal{S}(X)$ are retrieved. Each image in $\mathcal{S}(X)$ is expected to have the similar scene semantics and layout with image X . After that, the global class likelihood of pixels (global belief) are transferred from the statistics of pixel features in $\mathcal{S}(X)$. Concretely, among all the pixels in $\mathcal{S}(X)$, the semantic class of pixel X_i should match those pixels whose local representations are also close to X_i . Therefore, K pixels (from images within $\mathcal{S}(X)$) are firstly retrieved that are similar to X_i in the local representation space. Then the global belief is generated through a weighted voting based on the K retrieved pixels. Mathematically, it is derived in the following equation:

$$P_G^{\mathcal{S}(X)}(X_i, Y_j) = \frac{\sum_{k=1}^K \phi(X_i, X_k) \delta(Y(X_k) = Y_j)}{\sum_k \phi(X_i, X_k)} \quad (3.6)$$

where X_k is the k -th nearest neighbor pixel of X_i among all the pixels in $\mathcal{S}(X)$; $Y(X_k)$ is the ground truth label for pixel X_k ; $\delta(Y(X_k), Y_j)$ is an indicator function; $\phi(X_i, X_k)$ measures the similarity between X_i and X_k , which is defined over spatial and feature space:

$$\phi(X_i, X_j) = \exp(-\alpha \|x_i - x_j\|) \exp(-\gamma \|z_i - z_j\|) \quad (3.7)$$

where $x_i = F(X_i)$ denotes the CNN pixel feature for X_i , z_i is the normalized coordinate along the image height axis and α, γ controls the belief exponential falloff.

Meanwhile, as small-size object classes (e.g. ‘bird’ in the sky, ‘boat’ in the sea, etc) make negligible contribution to the global scene semantics, they may not appear in its nearest global exemplar images $\mathcal{S}(X)$. Thus, the global belief will be highly skewed to

² J is the number of regions. In our experiments, the image is divided into rectangular regions in a 2-layer spatial pyramid fashion [66].

frequent classes, which potentially harms the final class likelihood of rare classes according to the integration rule in Equation 3.4. To address this issue, we introduce an auxiliary pixel transfer set $\mathcal{A}(X)$. In detail, we first find the rare-class pixels whose quantities are below K (same value as in Equation 3.6) in $\mathcal{S}(X)$, and then augment the corresponding rare-class pixels until their quantities reach K . More specifically, $\mathcal{A}(X)$ is a set of rare-class pixels, and it is derived by randomly sampling the desired quantity of pixels from images outside $\mathcal{S}(X)$. Thus, the final transfer set is the combination of $\mathcal{S}(X)$ and $\mathcal{A}(X)$, and the quantity of every rare-class pixel is at least K . Thus, the global belief is expected to preserve the salient objects in the scene. Our non-parametric global belief estimation is reminiscent of popular label transfer works[91, 1, 92, 2, 93], two differences need to be highlighted:

- Instead of adopting hand-engineered low-level local and global features, we use more discriminative and compact features learned from CNN for label transfer.
- Our non-parametric model works as global scene constraints for local pixel features. Generally, small size retrieval images are sufficient to define the scene semantic and layout. However, previous works have to seek large retrieval set to cover all the possible semantic classes.

3.4.2 CNN-Metric

As shown in Equation 3.6, the estimation of global belief $P_G^{\mathcal{S}(X)}(X_i, Y_j)$ is highly dependent on the distance metric between two pixel features. However, our features are learned by optimizing pixel/patch classification accuracy, while do not take distance metric into consideration. Therefore we propose to learn a large-margin based metric to mitigate the inaccurate class likelihood estimation for rare classes (Figure 3.4). In detail, the Mahalanobis metric $M = W^T W$ is learned by minimizing the loss function, which is formally

Figure 3.4: Graphical illustration of the effect of our large margin based metrics. Due to the highly imbalanced data distribution, the nearest neighbors of the testing feature (triangle) are dominated by imposter classes (rectangle). After the metric transformation, the imposters stay far away from it. Thus, their contribution is significantly attenuated in the global belief estimation.

written as:

$$L = \frac{\lambda}{2} \|W\|^2 + \frac{1}{2N} \sum_{i,j} g(x_i, x_j) \quad (3.8)$$

$$g(x_i, x_j) = \max(0, 1 - \ell_{i,j}(\tau - \|Wx_i - Wx_j\|^2))$$

where $\ell_{i,j}$ indicates whether two features have the same semantic label or not, and $\ell_{i,j} = 1$ if X_i and X_j are from the same class, or $\ell_{i,j} = -1$ otherwise; $\tau (> 1)$ is the margin and λ controls the effect of regularization; $x_i = F(X_i)$ is the feature representation for X_i and N is the number of features. The objective function would enforce the pixel features from the same semantic class to be close and stay within the ball with radius $1 - \tau$, and enforce data from different classes to be far away from each other by at least $1 + \tau$. The graphical illustration of the metrics is depicted in Figure 3.4.

Instead of simply learning a metric based on the extracted CNN features, we further replace the softmax layer with our metric learning layer, so that the feature extraction parameters can also be adapted. We replace the softmax layer of previous CNN (CNN-softmax) with a fully connected layer parameterized by W (or more layers to learn non-linear metrics [125]) and fix the biases to be zero, which serves as a Mahalanobis metric ($M = W^T W$). We call the new network CNN-metric. These two networks do not share

any parameters except that the feature extraction parameters of CNN-metric are initiated from the corresponding layers of CNN-softmax. The errors are back propagated through the chain rule, and $\frac{\partial L}{\partial W} \frac{\partial L}{\partial x_i}$ for the last layer are given in Equation 3.9.

$$\begin{aligned}
 \frac{\partial L}{\partial W} &= \lambda W + \frac{1}{N} \sum_{i,j} \zeta_{ij} \\
 \zeta_{ij} &= g'(c) \ell(i, j) (Wx_i - Wx_j)(x_i - x_j)^T \\
 \frac{\partial L}{\partial x_i} &= \frac{1}{N} \sum_{i,j} g'(c) (W^T \ell(i, j) (Wx_i - Wx_j)) \\
 c &= 1 - \ell(i, j) (\tau - \|Wx_i - Wx_j\|^2) \\
 x_i &= F(X_i) \\
 g'(c) &= \begin{cases} 0, & c \leq 0 \\ 1, & c > 0 \end{cases}
 \end{aligned} \tag{3.9}$$

We adopt the Stochastic Gradient Descent (SGD) to optimize the CNN-Metric. Considering that the quantity of patches is prohibitively huge, we also sample a fraction of patches during each epoch in the **Class Sampling** manner (Section 3.3). Other sampling methods are not appropriate in this scenario, as the imbalanced data distribution will result in the scarcity of training examples for some class pairs, which is expected to skew the learned metric mapping. Furthermore, due to the infeasibility of feeding the sampled patches to the network in a single propagation, the training data are divided to several batches, among which the proposed metric constraint is enforced to any class pairs.

3.5 Experiments

3.5.1 Evaluation Benckmarks

We evaluate our approach on two popular and challenging scene parsing benchmarks:

- SiftFlow [1]: It has 2688 images generally captured from 8 typical natural scenes. Every image has 256×256 pixels, which belongs to one of 33 semantic classes.

We use the training/testing (2488/200 images) split provided by [1] to conduct our experiments.

- Barcelona [2]: It consists of 14871 training and 279 testing images. The size of the images varies across different instances, and each pixel is labelled as one of the 170 semantic classes. Note that the class frequency distribution is more imbalanced than that in the SiftFlow dataset. Meanwhile, the scene categories of training images range from indoor to outdoors, whereas the testing images are only captured from the barcelona street. These issues pose Barcelona as an extremely challenging dataset.

Evaluation metrics. In this thesis, we report three categories of evaluation metrics. Let n_{ij} be the number of pixels of class i predicted to belong to class j , where there are n_{class} different semantic classes, and let $t_i = \sum_j n_{ij}$ be the total number of pixels of class i . We compute,

- Global Pixel Accuracy (**GPA**), which is the percentage of all correctly classified pixels. $\mathbf{GPA} = \sum_I n_{ii} / \sum_i t_i$.
- Average per-Class Accuracy (**ACA**), which is the mean of class-wise pixel accuracy. $\mathbf{ACA} = 1/n_{class} \sum_i n_{ii}/t_i$.
- mean Intersection over Union (**IOU**)³. $\mathbf{IOU} = 1/n_{class} \sum_i n_{ii}/(t_i + \sum_j n_{ji} - n_{ii})$.

3.5.2 Local Labeling Results

We first present the implementation details of training the parametric CNNs. We use Stochastic Gradient Descent (SGD) with momentum to train the CNN-M; During each training epoch, around 5×10^5 pixels are randomly sampled from training pixel pools; The learning rate is initialized to be 0.01, and it is decreased by 10 times after 20 epoches; The

³We does not report IOU score in this chapter, as the IOU metrics on the experimental datasets are extremely low. For the same reason, our comparing literatures also does not report IOU metrics as well.

Figure 3.5: Visualization of the learned convolution filters for the first layer of CNN-CS, CNN-TCS, CNN-GS and CNN-HS respectively. They are trained on the Barcelona datasets. Note the visual difference among the learned filters of different CNN-M. The figure is best viewed in color.

momentum is fixed to 0.9, and the batch size is set as 100. We train our CNN-M based on MatConvnet [126] toolbox.⁴ The reported results are based on the model trained in 35 epoches. Similar to previous literatures [47] [56] [127], each image is preprocessed by first subtracting the mean and then performing contrast normalization by dividing its variance. Besides, the threshold η that discriminates rare classes are empirically set to 5% and 1% for the SiftFlow and Barcelona datasets respectively.⁵ The learned convolutional filters for the first layer of CNN-M on the Barcelona dataset are shown in Figure 3.5.

Next, we evaluate the performance of CNN-M. Specifically, CNNs output their local belief maps, based on which maximum marginal inference is performed to output the label prediction maps. Table 3.1 lists the quantitative results. As shown, our CNN-M (GS, HS) achieves very competitive results that are comparable or even much better than Multiscale ConvNet [47] and Recurrent CNN [56]. For example, CNN-GS outperforms Multi-scale ConvNet and Recurrent CNN by a significant 8% and 10% margin respectively in terms of GPA on SiftFlow dataset. CNN-HS outperforms Multi-scale ConvNet by a dramatic 23.2% (4.6%) over GPA (ACA) on Barcelona dataset. In terms of the

⁴The code is publicly available under the homepage of authors.

⁵This value is chosen based on 85%-15% rule used in [94] to differentiate frequent and rare classes: the accumulated frequency of all the non-rare classes is around 85%

	SiftFlow		Barcelona	
	GPA	ACA	GPA	ACA
Multiscale ConvNet [47]	67.9%	45.9%	37.8%	12.1%
Recurrent CNN (67×67) [56]	65.5%	20.8%	N/A	N/A
CNN-asymmetric [48]	42.4%	38.4%	20.0%	13.3%
CNN-GS	75.4%	30.2%	68.5%	11.4%
CNN-CS	70.9%	42.6%	24.7%	18.4%
CNN-TCS	7.91%	38.7%	6.33%	16.8%
CNN-HS	74.7%	39.4%	61.0%	16.7%
CNN-Ensemble	75.3%	44.8%	61.3%	19.5%
Ensemble CNN-GS	77.1%	32.0%	69.7%	11.5%
Ensemble CNN-CS	72.8%	43.7%	26.6%	20.0%
Ensemble CNN-TCS	8.33%	40.0%	7.53%	18.3%
Ensemble CNN-HS	76.4%	40.2%	63.2%	17.5%

Table 3.1: Quantitative Performance of different CNNs. Details of each method are elaborated in the text.

individual performance of CNN-M, **CNN-GS** achieves the best accuracy on global pixel accuracy (GPA), whereas its performance on average class accuracy (ACA) is not satisfactory. In contrast, **CNN-CS** claims the best ACA among all the single CNNs, which indicates that it predicts the semantic classes of pixels in a more equal manner. A favorable performance compromise is achieved by **CNN-HS**, which works considerably well on both GPA and ACA. **CNN-TCS** performs extremely poor on GPA as it ignores the frequent classes, it however achieves very competitive ACA. Importantly, CNN-TCS captures image statistics that are significantly disparate from other networks (CNN-GS, HS, CS): it performs the best to correctly recognize the rare-class pixels. Quantitatively, the CNN-Ensemble that excludes CNN-TCS only achieves 40.3% and 17.8% in terms of ACA on the SiftFlow and Barcelona dataset respectively, and their performance are boosted to 44.8% and 19.5% after including CNN-TCS. By fusing the complementary decisions from different CNN-M, **CNN-Ensemble** achieves the best performance tradeoff.

Furthermore, we train **Ensemble CNN-M** to compare their performance behaviour with CNN-Ensemble. In detail, the CNN-M in Ensemble CNN-M is trained with the image patches sampled from the identical method M. As manifested in Table 3.1, the Ensemble CNN-M significantly improves the local labeling performance over CNN-M. However, they optimize only one evaluation criterion (either GPA or ACA), and this phenomenon is more obvious in the severely imbalanced Barcelona dataset. In contrast, the CNN-Ensemble performs competitively excellently on both GPA and ACA. It's also important to note that CNN-HS behaves similarly with CNN-Ensemble and it produces very promising results on both GPA and ACA. However, the performance of Ensemble CNN-HS is significantly inferior to CNN-Ensemble in terms of ACA.

In the end, we discuss the issue of imbalanced class frequency distribution in scene parsing. In order to boost the recognition rates for infrequent classes, we train another **CNN-asymmetric** as in [48], whose log-loss is modulated by the inverse frequency of each class, thus the rare classes are effectively given more attention. In this case, the training data can be generated as in [48] by collecting all image patches (or via global sampling strategy). The corresponding result is reported in Table 3.1, which shows that CNN-asymmetric fails to achieve the desirable results as in object segmentation benchmarks [48]. This phenomenon can be explained from the following perspective. The class frequency distribution in the object segmentation task is not as imbalanced as in the scene parsing task ⁶. If the inverse frequency is used to scale the log-loss in this scenario, the scaled losses w.r.t the frequent classes are negligible. Consequently, the CNN-asymmetric performs poorly on GPA on the scene parsing benchmarks. In contrast, the sampling based CNNs (e.g. CNN-CS, CNN-HS, CNN-GS) achieve much better performance tradeoff, which positively elucidates the effectiveness of the proposed sampling strategy to address the class imbalance issue. More importantly, by feeding the networks with different sampled patches during the training phase, we are able to train

⁶Statistically, the frequency ratio between the most frequent and rare classes on the PASCAL VOC 2011 [48] and SiftFlow datasets (see Figure 3.3) are approximately 240 and 3.5×10^4 respectively.

	Dim	K=1	K=5	K=10
GIST [128]	512	74.0%	70.7%	68.3%
SIFT-SPM [66]	2100	76.5%	71.3%	69.1%
Global Feature (CNN-GS)	320	91.5%	88.3%	86.5%
Global Feature (CNN-HS)	320	88.5%	85.8%	85.2%
Global Feature (CNN-CS)	320	90.5%	84.7%	84.1%
Global Feature (CNN-TCS)	320	79.5%	73.9%	72.8%
Global Feature (CNN-Ensemble)	1280	91.5%	87.4%	86.7%
GT	165D	94.0%	91.0%	89.5%

Table 3.2: Average genuine matching percentage among their K-nearest neighbors for different global features. GT is the semantic feature pooled from ground truth label maps.

a number of complementary CNNs, and combine them to produce much stronger local prediction model (CNN-Ensemble).

3.5.3 Evaluation of Global Features

In this section, we demonstrate that the pooling operation of pixel features is capable of generating semantically consistent global features. To achieve this goal, we calculate the KNN matching score p - the average genuine matching percentage among their K nearest neighbors. It is Mathematically derived in the following equation: $p = \frac{\sum_i^N \sum_k^K \delta(I_i, NN(i, k))}{NK}$, where N is the number of test images, $NN(i, k)$ stands for the k -th nearest neighbor for image I_i , and $\delta(i, j)$ outputs value 1 if i and j are a genuine match, or 0 otherwise. A genuine matching image pair means that they belong to the identical semantic class. We test the global features on the SiftFlow dataset, as it provides the global scene label for each image.

Four global features are compared in our experiment: GIST [128] is a global summary of scene images that captures scene structure and layout; SIFT-SPM (GT) [66] is pooled from low-level local SIFT [51] (ground truth label map [1]) in a 3(2)-layer spatial pyramid. They are commonly used in scene classification and non-parametric label transfer

framework. GT is the ideal global semantic feature. As mentioned in Section 3.4, our global feature is pooled from the output of truncated CNN-M in a 2-layer spatial pyramid fashion. Euclidean distance is used to retrieve nearest neighbors for non-histogram features (GIST, Ours), and histogram intersection similarity measurement is applied for the rest histogram features (SIFT-SPM and GT).

The quantitative matching scores for different global features are listed in Table 3.2. As demonstrated, our global feature is more likely to group semantically relevant images together. Meanwhile, among all of the global features(CNN-M), CNN-GS performs the best and CNN-TCS works the worst, as the scene semantics of outdoor images are mostly determined by the appearance of frequent classes. It’s worth noting that the concatenation of global features from different CNN-M fails to outperform CNN-GS. Hence, only global feature (CNN-GS) is used subsequently to retrieve the nearest exemplars. It’s also interesting to observe that the retrieval performance based on GT features are imperfect, which implies that different scenes can have very similar building blocks. For example, ‘inside city’ and ‘street’ scenes are dominated by sky and building pixels. We believe that the quality of nearest neighbor retrieval directly determines the correctness of global belief. Therefore, our global feature is expected to benefit other label transfer works as well. Some qualitative examples are shown in Figure 3.6, in which the retrieved nearest exemplar images have very similar scene layout.

3.5.4 Global Labeling Results

We first present the implementation details for our non-parametric model. The non-overlapping patches are adopted as label transfer units, within which labels are assumed to be identical. Specifically, as our CNN-M has three pooling layers, the feature extractor (truncated CNN-M) can be regarded as sliding the images with a stride of 8. In other words, the dimension of the output feature tensor F is $\frac{1}{8}$ of original image size: one feature in F corresponds to a 8×8 image patch. In our experiments, we only estimate

the class likelihood for each feature in F , implicitly assuming that class labels within the 8×8 regions are the same.⁷

The global belief is calculated by Equation 3.6, in which $|\mathcal{S}(X)|$ (size of nearest exemplar images) and K (size of nearest pixel/patch neighbors) are empirically set to be 5 and 200 respectively. As manifested in Table 3.2, 5 images are sufficient to correctly define the scene semantics. However, as many images are not fully annotated in the Barcelona dataset, a larger retrieval image set is used ($|\mathcal{S}(X)| = 100$). α and γ in Equation 3.7 are empirically set to 15 and 5 respectively. We find that the results are quite robust to value changes of α and γ . In our experiments, we observed marginal performance disparity when α ranges from 10 to 15, and γ ranges from 0 to 5. To fine-tune the CNN-metric, 1×10^4 patches are sampled for each class in each epoch (which results in 3.3×10^5 and 1.7×10^6 training patches on the SiftFlow and Barcelona dataset respectively), and 2000 patches are used to be a training batch. λ and τ in Equation 3.8 makes marginal difference to the performance, and they are fixed to 0.01 and 3 respectively. The learning rate is initialized to 10^{-3} and decays exponentially with the rate of 0.9. The reported results are obtained under the models learned in 20 epoches.

Similarly, the global labeling results are obtained through performing maximum marginal inference over global beliefs. The quantitative results on the SiftFlow and Barcelona dataset are listed in Table 3.3 and 3.4 respectively. From which, we clearly observe that our simple non-parametric model achieves very promising results that are comparable or even better than most complicated label transfer counterparts. For example, we reach 78.8% (39.6%) in terms of GPA (ACA) on the SiftFlow dataset, whereas SuperParsing [2] and graph transfer [96] only achieves 76.9% (29.4%) and 78.4% (25.7%) respectively. Moreover, our global labeling alone already achieves state-of-the-art results on the challenging Barcelona dataset. We attribute the performance superiority to the

⁷This is not the optimal setting, as it oversmooths the label prediction maps. However, it is fast and easy to implement, and we don't expect that the global belief to preserve the boundary information as it simply reflects the global scene prior. Hence, it's a reasonable compromise.

	Global (GPA)	Class (ACA)
SuperParsing [2]	76.9%	29.4%
Liu et al. [1]	74.8%	N/A
Gould et al. [96]	78.4%	25.7%
Singh et al. [92]	79.2%	33.8%
Tighe et al. [3]	78.6%	39.2%
Yang et al. [94]	79.8%	48.7%
Tung et al. [129]	79.9%	49.3%
George et al. [4]	81.7%	50.1%
Raw Multiscale ConvNet [47]	67.9%	45.9%
Raw Multiscale ConvNet [47] + Cover	72.3%	50.8%
Raw Multiscale ConvNet [47] + Cover	78.5%	29.4%
Plain CNN (133×133) [56]	76.5%	30.0%
Recurrent CNN (133×133) [56]	77.7%	29.8%
Gatta et al. [130]	78.7%	32.1%
Long et al. [21] (ImageNet Pretrain)	85.2%	51.7%
Local Labeling (CNN-Ensemble)	75.3%	44.8%
Global Labeling	78.7%	36.2%
Global Labeling (Metric)	78.8%	39.6%
Integration model	81.0%	44.6%
Integration model (Metric)	81.2%	45.5%

Table 3.3: Quantitative performance of different methods on the SiftFlow datasets.

highly discriminative CNN features which we work with in the non-parametric transfer model.

Furthermore, as evidenced by Table 3.3 and 3.4, the learned metric is capable of improving the quality of global beliefs for rare classes. As illustrated in Figure 3.4, the learned metric is expected to shrink the distance of pixel features between identical classes and enlarge the distance between disparate classes. Consequently, it benefits the estimation of global beliefs for infrequent classes. In our experiments, we do observe the desirable average class accuracy (ACA) boost by incorporating metric tuning on both datasets.

	Global (GPA)	Class (ACA)
SuperParsing [2]	66.9%	7.6%
Raw Multiscale ConvNet [47]	37.8%	12.1%
Raw Multiscale ConvNet [47] + Cover	46.4%	12.5%
Raw Multiscale ConvNet [47] + Cover	67.8%	9.5%
Local Labeling (CNN-Ensemble)	61.3%	19.5%
Global Labeling	68.1%	13.1%
Global Labeling (Metric)	68.7%	14.1%
Integration Model	69.7%	16.8%
Integration Model (Metric)	70.3%	17.2%

Table 3.4: Quantitative performance of different methods on the Barcelona dataset.

3.5.5 Integration Labeling Results

The integration model applies Equation 3.4 to integrate the local and global beliefs, which yields the un-normalized class likelihood of pixels. Based on which, the final prediction map is produced. Table 3.3, 3.4 list the quantitative performance on the SiftFlow and Barcelona dataset respectively. As shown, our integration model is able to take advantage of both models, therefore it outputs more reliable label maps quantitatively and qualitatively. On one hand, *it dramatically boosts the global pixel accuracy (GPA) compared to the local labeling (CNN-Ensemble): 5.9% and 9.0% GPA improvement on the SiftFlow and Barcelona dataset respectively.* These results elucidate that a higher quality labeling prediction map can be obtained by enforcing a global scene constraint to the local classification. On the other hand, *it also tremendously enhances the average class accuracy (ACA) compared to the global labeling: 9.3% and 4.1% ACA improvement on the Siftflow and Barcelona dataset respectively.* These results indicate that some object classes are ignorable in a global view (e.g. a small bird flies in broad sky), and the integration of local cues is akin to delineating objects in a global scene image. A number of qualitative examples are demonstrated in Figure 3.6.

In comparison with other representation networks [47] [56], our integration model

outperforms them by a large margin. Thus, enforcing a global scene constraint to local classification is a promising solution to alleviate *local ambiguities*. Furthermore, we compare our integration model with state-of-the-art counterparts. As listed in Table 3.3, our method achieves very competitive results on the SiftFlow dataset. It’s important to note that [4] uses 20 types of low-level features and further augments it with Fisher Vector (FV) [131] descriptor (based on the SIFT feature) to represent each image superpixel. In contrast, we only use very compact (64-dimension) features learned from CNNs. In addition, [21] adopts the very deep CNNs [13] pretrained on the large-scale ImageNet dataset [14] to generate the local features, whereas we utilize much shallower CNNs, which are trained with image patches only from the target dataset. Meanwhile, Table 3.4 clearly manifests that our method achieves the new best results on the more challenging Barcelona dataset, which significantly outperforms the previous state-of-the-arts.

3.5.6 Analysis of Local and Global Beliefs

In this section, we first compare the characteristics of local and global beliefs. By looking into the local/global labeling results quantitatively and qualitatively, we notice that the global labeling is more likely to output globally consistent label maps, while the local labeling focuses on discriminating local regions. Specifically, the global belief prioritizes the background classes, whereas the local belief works excellently on differentiating locally distinct object classes (especially small size classes). Hence, the global belief defines the global scene prior, while local belief preserves the locality information. Moreover, the reciprocity of local and global beliefs necessitates the competitive performance of our integration model. A glimpse of these properties are depicted in the qualitative labeling examples of Figure 3.6. Meanwhile, It is interesting to see that the global labeling outperforms local labeling (CNN-GS), which illuminates the significance of global context for local classification.⁸

⁸Under this situation, these two models are working on the same pixel features, which are generated by truncated CNN-GS.

Figure 3.6: Qualitative labeling examples on the SiftFlow dataset (best viewed in color). In each row, we show input images, top three nearest exemplar images, local labeling maps (output by the parametric model - CNN-Ensemble), global labeling maps (output by the non-parametric model), final label prediction maps (output by the integration model) and their ground truth maps respectively. The numbers outside and inside parentheses are global pixel accuracy (GPA) and average class accuracy (ACA) respectively. The last row shows an example, where the *local ambiguity* cannot be removed when their aggregated global features fail to reveal the true scene semantics.

	Frequency	CNN-Ensemble	Integration Model
sky	27.1%	93.5%	96.7%
building	20.2%	80.6%	88.0%
tree	12.6%	74.8%	84.7%
mountain	12.4%	71.5%	80.3%
road	6.93%	76.8%	85.8%
sea	5.6%	59.7%	75.0%
field	3.64%	33.1%	37.6%
car	1.6%	80.5%	78.7%
sand	1.41%	33.6%	37.5%
river	1.37%	44.0%	50.1%
plant	1.33%	27.5%	7.56%
grass	1.22%	75.2%	72.6%
window	1.07%	45.1%	33.8%
sidewalk	0.89%	60.2%	53.7%
rock	0.85%	23.8%	13.0%
door	0.26%	36.8%	40.2%
fence	0.24%	44.5%	44.6%
person	0.23%	30.9%	44.2%
staircase	0.18%	45.4%	24.6%
awning	0.11%	7.54%	14.5%
sign	0.11%	33.4%	50.3%
boat	0.06%	2.22%	3.05%
crosswalk	0.05%	85.4%	74.1%
bridge	0.04%	18.8%	20.4%
pole	0.04%	2.43%	7.1%
balcony	0.03%	34.4%	29.9%
bus	0.03%	0.05%	0.05%
streetlight	0.02%	15.1%	2.36%
sun	0.01%	93.3%	86.0%
bird	≈ 0	13.2%	28.3%
GPA	-	75.3%	81.2%
ACA	-	44.8%	45.5%

Table 3.5: Per-class accuracy comparison of different models on the SiftFlow dataset. The statistics of class frequency is obtained in test images.

We further investigate the per-class accuracy changes of the parametric model after it integrates with the non-parametric model. Table 3.5 shows the quantitative results on the SiftFlow dataset: the integration model boosts the accuracy significantly for frequent classes, while slightly washes away some rare “object” classes. In detail, the global belief is more helpful for classes which are more stable in positions, and large-size classes are preferred because the target classes to be included in the nearest exemplars. Overall, our integration model is able to achieve very competitive average class accuracy (ACA), and in the same time dramatically improve the qualitative labeling results. As evidenced by Table 3.1 and 3.4, similar results are also observed on more challenging Barcelona datasets.

3.6 Conclusion

in this chapter, we first present a very effective parametric model - **CNN-Ensemble** - for local classification. The CNN components in the CNN-Ensemble are trained from image patches which are very different in the form of class frequency distribution. Therefore, each CNN component learns nonidentical visual patterns, and their decision fusion gives rise to more accurate local beliefs. Then, we alleviate the notorious *local ambiguity* problem by introducing a global scene constraint, which is mathematically achieved by adding a global energy term to the labeling energy function, and it is practically estimated in a non-parametric framework. Furthermore, A large margin based CNN metric learning method is also proposed for better global belief estimation. The final class likelihood of pixels are obtained by integrating local and global cues. The outstanding quantitative and qualitative results on the challenging SiftFlow and Barcelona datasets illuminate the effectiveness of our methods.

Chapter 4

DAG-Recurrent Neural Networks

4.1 Introduction

Scene segmentation refers to labeling each pixel to be one of the semantic categories that include both stuff (e.g. sky, grass, etc) and object classes (e.g. tree, person, etc). In order to parse scene images robustly, effective contextual modeling is more demanding. To elaborate, beyond modeling object-object interactions (in object segmentation), a scene parser needs to capture stuff-stuff, stuff-object dependencies as well. Besides, understanding global scene semantics and structure is also essential to disambiguate the appearance nuances. For example, local features for ‘sand’ and ‘road’ are indistinguishable in some cases (c.f. *local ambiguity* in Chapter 3), however, they can be differentiated easily after the global scene (e.g. ‘coast’) is revealed. In the mean time, the identities of small-size objects should also be inferred easier after they are well contextualized.

In this chapter, we propose DAG-RNN to capture the rich contextual dependencies over image regions. To be specific, we locally connect features (in feature maps) to form a graph (UCG, which refers to the undirected graphs in which there exists a closed walk with no repetition of edges.) Due to the loopy property of UCGs, RNNs are not directly applicable to UCG-structured images. Thus, we decompose the UCG to several directed acyclic graphs (DAGs). In other words, an UCG-structured image is represented by the combination of several DAG-structured images. Then, we develop DAG-RNN, a

generalization of RNNs [16][17], to explicitly pass local context based on the directed graphical structure. By doing this, contexts are explicitly propagated and encoded into feature maps.

We place DAG-RNN on top of a pre-trained CNN, which gives rise to the backbone of our segmentation network that differentiates FCN [21] in the aspect of context aggregation. Specifically, DAG-RNN replace convolution kernels (in FCN) to perform context aggregation. There are multiple appealing properties by doing this. First, DAG-RNN entails significantly fewer parameters than CNN. In addition, it requires significantly less computations to complete the feed-forward pass. Therefore, DAG-RNN is more favorable to be potentially applied in resource-constrained embedded devices. Most importantly, DAG-RNN is more effective in context aggregation, which demonstrates significantly better scene parsing performance than FCNs (including DeepLab [42], ParseNet [33]).

In order to address the severely imbalanced class frequency distribution, we propose a novel class-weighted loss to train the segmentation network. Inspired by the ‘hybrid’ sampling in [132][50], the class-weighted function distributes reasonably higher attention weight to infrequent classes based on their rareness magnitude. Thus, those infrequent class pixels are expected to be learned faster comparing to frequent classes. Our experiments show that the proposed class-weighted loss is a very economic as well as an effective way to boost the parsing accuracies of rare classes.

We test the segmentation network on three scene segmentation datasets (Sift Flow [1], PASCAL Context [11] and COCO Stuff [18]), where it achieves very impressive parsing performance. Besides, we quantitatively and qualitatively compare and discuss how DAG-RNN and CRF [41] achieve contextual modeling. Meanwhile, ablation studies and exploratory experiments on object segmentation dataset (PASCAL VOC [20]) are carried out to discuss several important components in DAG-RNN.

Figure 4.1: Our basic segmentation network essentially consists of three temporal-consecutive functional modules: local representation, context aggregation and feature map upsampling. Below, an example of the proposed segmentation network is demonstrated that subsumes convolutional network, DAG-RNN and deconvolution network. C denotes the cardinality of semantic classes considered in the target dataset. Shortcut lines (connections) symbolize that the outcoming feature maps (e.g. $28 \times 28 \times 512$) are firstly locally predicted and then summed with preceding (upsampled) predictions.

4.2 Segmentation Network

As demonstrated in Fig. 4.1, we introduce the basic segmentation network with three functional modules: (1) local region representation, (2) context aggregation, and (3) feature map upsampling. To elaborate,

- (1) Local region representation module outputs compact and discriminative representations for pixels. An instance can be a convolutional neural network (CNN, with pooling layers) [21] [132][42][34]. The sub-sampling layers in CNN can be interpreted as the sliding stride during the feature extraction, which therefore results in the down-sampled spatial resolution of feature maps.
- (2) Context aggregation module engages extra useful contexts to local features. It is important to note that this module does not change the spatial dimensionality of feature maps, whereas it aims to inject informative context to enhance the representative capability of feature maps. Convolution kernels [21] (or its variants, e.g.

dilated (atrous) convolution kernels [42][34]), recursive neural network [32], global pooling layer [33] have been adopted to achieve context aggregation in previous literatures.

- (3) Feature map upsampling is to recover the details in down-sampled feature maps, which enables the segmentation network to generate high-resolution label prediction maps. The deconvolution network [38] (inverse convolution [21]) or simple bilinear kernels are commonly used to achieve this functionality.

A detailed example of segmentation network is depicted in Figure 4.1. Optionally, we can add another functional module akin to fully connected CRF [42][43] to further improve the visual quality of the predicted maps.

4.3 Context aggregation via convolution kernels

Context aggregation module (layer) is designed to encode wider range of contextual information into local features, namely it spans the field of views (receptive fields) of feature maps as well. In FCN [21], plain CNN (i.e. convolution kernels) are adopted. As an example in Fig. 4.2, an image is represented as a feature tensor with dimensionality of $8 \times 8 \times \mathbf{d}$. The convolution kernel with size $7 \times 7 \times \mathbf{d} \times \mathbf{m}^1$ is used to aggregate neighborhood context. The feature highlighted with magenta color in Fig. 4.2 (b) is able to incorporate wide-range context that covers nearly the entire image. In this way, its representative capability is improved. However, there are obvious drawbacks by doing this.

First, plain CNN consumes large quantity of parameters. For example, it entails more than 100 million parameters in FCNs [21] that accounts for nearly 70% of network parameters. Not surprisingly, it requires extensive FLOPs (FLoating-point OPerations) to

¹height $\times$ width $\times$ #channels $\times$ #kernels, hereinafter, convolution kernels are displayed in this format.

Figure 4.2: Illustration of context aggregation. (a), feature tensor (with size of $8 \times 8 \times d$) for the given scene image, and each square denotes one feature vector in the feature tensor; (b), convolution kernel (with the size of $7 \times 7 \times d \times m$) is used to aggregate context for local features. The two dashed squares symbolize the field of views of two corresponding local features (highlighted with blue/magenta color); (c), local features are connected in an 8-neighborhood UCG, under which any pair of local features can be mutually communicable; and (d), one of its induced DAG in the southeastern (SE) direction. Context aggregation is achieved by propagating local context based on such DAG topology.

complete the computation of this layer as well. Therefore, such segmentation network is less desirable to be applied to devices where memory storage and computation resources are both limited. Second, it's effort-demanding to tune the optimal network architecture for images with different resolutions. In order to encode effective wide field of context (e.g. entire image), small conv kernels (context aggregation module in FCN) are sufficient for low-resolution inputs, whereas such configuration cannot expand relatively large receptive fields for high-resolution images.

4.4 DAG-Recurrent Neural Networks

In this chapter, we propose to adopt DAG-RNN to perform context aggregation. In a nutshell, local context can be propagated to the entire image with DAG-RNN, so feature maps essentially have a contextual view of the full image regardless of image resolutions. In addition, DAG-RNN is a light model that entails very few parameters. These properties make DAG-RNN an appealing choices for context aggregation module. Subsequently, we introduce the mechanism of DAG-RNN, and elaborate how it is used

to aggregate context.

4.4.1 Chain-RNN Revisited

A recurrent neural network (RNN) is a class of artificial neural network that has recurrent connections, which equip the network with memory. In this chapter, we focus on the Elman-type network [133]. Specifically, the hidden layer $\mathbf{h}^{(t)}$ in RNNs at time step t is expressed as a non-linear function over current input $\mathbf{x}^{(t)}$ and hidden layer at previous time step $\mathbf{h}^{(t-1)}$. The output layer $\mathbf{y}^{(t)}$ is connected to the hidden layer $\mathbf{h}^{(t)}$. Mathematically, given a sequence of inputs $\{\mathbf{x}^{(t)}\}_{t=1:T}$, an Elman-type RNN operates by computing the following hidden and output sequences:

$$\begin{aligned}\mathbf{h}^{(t)} &= f(U\mathbf{x}^{(t)} + W\mathbf{h}^{(t-1)} + b) \\ \mathbf{y}^{(t)} &= g(V\mathbf{h}^{(t)} + c)\end{aligned}\tag{4.1}$$

where U, W are weight matrices between the input and hidden layers, and among the hidden units themselves, while V is the output matrix connecting the hidden and output layers; b, c are corresponding bias vectors and $f(\cdot), g(\cdot)$ are element-wise nonlinear activation functions. The local information $\mathbf{x}^{(t)}$ is progressively stored in the hidden layers by applying Equation 4.1. In other words, the contextual information (the summarization of past sequence information) is explicitly encoded into local representation $\mathbf{h}^{(t)}$.

Training a RNN can be achieved by optimizing a discriminative objective with a gradient-based method. Back Propagation through time (BPTT) [134] is usually used to calculate the gradients. This method is equivalent to unfolding the network in time and using back propagation in a very deep feed-forward network except that the weights across different time steps (layers) are shared.

4.4.2 DAG-RNN

The aforementioned RNN is designed for chain-structured sequential data (e.g. sentences or speeches), where temporal dependency is modeled. However, the connectivity structure of image units are beyond chain. In other words, traditional chain-structured RNNs

are not suitable for images. For instance, we can reshape the feature tensor $\mathbf{x} \in \mathbb{R}^{h \times w \times d}$ to $\hat{\mathbf{x}} \in \mathbb{R}^{(h \cdot w) \times d}$, and generate its chain representation by connecting contiguous elements in $\hat{\mathbf{x}}$. Such a structure loses spatial arrangement of image units, as two adjacent local features in image plane may not necessarily be neighbors in the chain. The graphical representations that respect the 2-D neighborhood system are more plausible solutions, and they are pervasively adopted in probabilistic graphical models (PGM). Therefore in this work, undirected cyclic graphs (UCG), an example is shown in Fig. 4.2 (c)) are used to represent the connectivity structure of image units.

Due to the loopy structure of UCGs, they are unable to be unrolled to an acyclic processing sequence. Therefore, RNNs are not directly applicable to UCG-structured images. To address this issue, we approximate the topology of UCG by a combination of several directed acyclic graphs (DAGs), each of which is applicable for our proposed DAG-RNN (one of the induced DAGs is depicted in Fig. 4.2 (c)). Namely, an UCG-structured image is represented as the combination of a set of DAG-structured images. We now start introducing the detailed mechanism of our DAG-RNN here, and later elaborate how they are applied to UCG-structured images in the next section.

We first assume that the topology of an image I is represented as a DAG $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_i\}_{i=1:N}$ is the vertex set and $\mathcal{E} = \{e_{ij}\}$ is the arc set (e_{ij} denotes an arc from v_i to v_j). The topology of the hidden layer $\mathbf{h}$ follows the same topology as $\mathcal{G}$. Therefore, a forward propagation sequence can be generated by traversing $\mathcal{G}$, on the condition that one node should not be processed until all its predecessors are processed. The hidden activation $\mathbf{h}^{(v_i)}$ is represented as a nonlinear function over its local input $\mathbf{x}^{(v_i)}$ and the summarization of hidden representation of its predecessors. In detail, the forward operation of DAG-RNN is calculated by the following equations:

$$\begin{aligned} \hat{\mathbf{h}}^{(v_i)} &= \sum_{v_j \in \mathcal{P}_{\mathcal{G}}(v_i)} \mathbf{h}^{(v_j)} \\ \mathbf{h}^{(v_i)} &= f(U\mathbf{x}^{(v_i)} + W\hat{\mathbf{h}}^{(v_i)} + b) \\ \mathbf{o}^{(v_i)} &= g(V\mathbf{h}^{(v_i)} + c) \end{aligned} \tag{4.2}$$

where $\mathbf{x}^{(v_i)}$, $\mathbf{h}^{(v_i)}$, $\mathbf{o}^{(v_i)}$ are the representations of input, hidden and output layers located at v_i respectively, $\mathcal{P}_{\mathcal{G}}(v_i)$ is the direct predecessor set of vertex v_i in the graph $\mathcal{G}$, $\hat{\mathbf{h}}^{(v_i)}$ summarizes the information of all the predecessors of v_i . Note that the weight matrix W in Equation 4.2 is shared across all predecessor vertexes in $\mathcal{P}_{\mathcal{G}}(v_i)$. We may learn a specific weight matrix W for each predecessor when vertexes (except source and sink vertex) if the DAG $\mathcal{G}$ have a fixed number of predecessors.

The derivatives are computed in the backward pass, and each vertex is processed in the reverse order of forward propagation sequence. Specifically, to derive the gradients at v_i , we look at equations (besides Equation 4.2) that involve $\mathbf{h}^{(v_i)}$ in the forward pass:

$$\begin{aligned} \forall v_k \in \mathcal{S}_{\mathcal{G}}(v_i) \\ \mathbf{h}^{(v_k)} &= f(U\mathbf{x}^{(v_k)} + W\mathbf{h}^{(v_i)} + W\tilde{\mathbf{h}}^{(v_k)} + b) \\ \tilde{\mathbf{h}}^{(v_k)} &= \sum_{v_j \in \mathcal{P}_{\mathcal{G}}(v_k) - \{v_i\}} \mathbf{h}^{(v_j)} \end{aligned} \quad (4.3)$$

where $\mathcal{S}_{\mathcal{G}}(v_i)$ is the direct successor set for vertex v_i in the graph $\mathcal{G}$. It can be inferred from Equation 4.2, 4.3 that the errors backpropagated to the hidden layer ($d\mathbf{h}^{(v_i)}$) at v_i have two sources: direct errors from v_i ($\frac{\partial \mathbf{o}^{(v_i)}}{\partial \mathbf{h}^{(v_i)}}$), and summation over indirect errors propagated from its successors ($\sum_{v_k \in \mathcal{S}_{\mathcal{G}}(v_i)} \frac{\partial \mathbf{o}^{(v_k)}}{\partial \mathbf{h}^{(v_k)}} \frac{\partial \mathbf{h}^{(v_k)}}{\partial \mathbf{h}^{(v_i)}}$). The derivatives at v_i can then be computed by the following equations:

$$\begin{aligned} \Delta V^{(v_i)} &= g'(\mathbf{o}^{(v_i)})(\mathbf{h}^{(v_i)})^T \\ d\mathbf{h}^{(v_i)} &= V^T g'(\mathbf{o}^{(v_i)}) + \sum_{v_k \in \mathcal{S}_{\mathcal{G}}(v_i)} W^T d\mathbf{h}^{(v_k)} \circ f'(\mathbf{h}^{(v_k)}) \\ \Delta W^{(v_i)} &= \sum_{v_k \in \mathcal{S}_{\mathcal{G}}(v_i)} d\mathbf{h}^{(v_k)} \circ f'(\mathbf{h}^{(v_k)})(\mathbf{h}^{(v_i)})^T \\ \Delta U^{(v_i)} &= d\mathbf{h}^{(v_i)} \circ f'(\mathbf{h}^{(v_i)})(\mathbf{x}^{(v_i)})^T \end{aligned} \quad (4.4)$$

where $\circ$ denotes the Hadamard product. By abuse of notation, $g'(\cdot) = \frac{\partial L}{\partial \mathbf{o}} \frac{\partial \mathbf{o}}{\partial g}$ is the derivative of loss function L with respect to the output function g , and $f'(\cdot) = \frac{\partial \mathbf{h}}{\partial f}$.² It

²Here, g (and f) are composition functions. For instance, $\mathbf{o} = g(p(\mathbf{h}))$ where $p(\mathbf{h}) = V\mathbf{h} + c$. $\frac{\partial \mathbf{o}}{\partial g}$ actually refers to $\frac{\partial \mathbf{o}}{\partial p(\mathbf{h})}$.

Figure 4.3: An example illustrates the decomposition of an UCG to four complimentary DAGs: $\mathcal{U} = \{\mathcal{G}_1, \dots, \mathcal{G}_4\}$. Note that any vertex pair can still be mutually communicable in the resulting set of DAGs. For example, local information of the yellow vertex can be routed to magenta vertex via $\mathcal{G}_1$, meanwhile yellow vertex can glean features from magenta vertex via propagating path in $\mathcal{G}_4$.

is the second term of $d\mathbf{h}^{(v_i)}$ in Equation 4.4 that enables DAG-RNN to propagate local information, which behaves similarly to the message passing [135] in probabilistic graphic models.

4.4.3 Decomposition

We decompose the UCG $\mathcal{U}$ to a set of DAGs $\mathcal{G}^{\mathcal{U}} = \{\mathcal{G}_1, \dots, \mathcal{G}_d, \dots\}$. Hence, the UCG-structured image is represented as the combination of a set of DAG-structured images. Next, DAG-RNN is applied independently to each DAG-structured image, and the corresponding hidden feature map (tensor) $\mathbf{h}_d$ is generated. The aggregation of these independent feature maps gives rise to the final hidden layer $\mathbf{h}$. These operations can be mathematically expressed as follows:

$$\begin{aligned} \mathbf{h}_d^{(v_i)} &= f(U_d \mathbf{x}^{(v_i)} + \sum_{v_j \in \mathcal{P}_{\mathcal{G}_d}(v_i)} W_d \mathbf{h}_d^{(v_j)} + b_d) \\ \hat{\mathbf{o}}^{(v_i)} &= g\left(\sum_{\mathcal{G}_d \in \mathcal{G}^{\mathcal{U}}} V_d \mathbf{h}_d^{(v_i)} + c\right) \end{aligned} \quad (4.5)$$

where U_d, W_d, V_d and b_d are weight matrices and bias vector for the DAG $\mathcal{G}_d$, $\mathcal{P}_{\mathcal{G}_d}(v_i)$ is the direct predecessor set of vertex v_i in $\mathcal{G}_d$. This decomposition strategy is reminiscent of the tree-reweighted max-product algorithm (TRW) [136], which represents the problem on the loopy graphs as a convex combination of tree-structured problems.

We consider the following criterion for the decomposition: the combination of DAG set $\mathcal{G}^{\mathcal{U}} = \{\mathcal{G}_1, \dots, \mathcal{G}_d, \dots\}$ should be topologically equivalent to UCG $\mathcal{U}$. In other words, any two vertexes can be mutually reachable in the resulting set of DAGs. In this scenario, local information can be routed to anywhere in the image, which is of great essentiality to achieve robust high-level scene understanding. As exemplified in Figure 4.3, we use the four context propagation directions (southeast, southwest, northwest and northeast) suggested by [16][112] to decompose the UCG in our experiment.

4.4.4 Segmentation network with DAG-RNN

Multi-scale features are essential to boost the performance in vision tasks. This is also true for scene segmentation. Lower-level features preserve finer-grained spatial information which is beneficial for recognizing small-size objects as well as localizing boundaries. Higher-level features are more translation invariant, and they are able to provide rough yet robust high-level understanding of the structure of scenes. Therefore, it's advantageous to combine their predictions towards accurate scene segmentation. In practice, we add branches from lower-level convolutional layers, and fuse their outputs with DAG-RNN.

The architecture of segmentation network is illustrated in Fig. 4.1. The network is end-to-end trainable, and it takes input as raw RGB images with any size. It outputs the label prediction maps with the same size of inputs. We adopt the average weighted cross entropy loss to train the segmentation network. It is formally written as:

$$L = -\frac{1}{N} \sum_{v_i \in I} \sum_{j=1}^c \mathbf{w}_j \mathbf{y}_j^{(v_i)} \log(\tilde{\mathbf{o}}_j^{(v_i)})$$

$$\tilde{\mathbf{o}} = \text{deconv}(\hat{\mathbf{o}})$$
(4.6)

where N is the number of image units in image I ; $\mathbf{w}$ is the class weight vector, in which $\mathbf{w}_j$ stands for the weight for class j ; $\mathbf{y}^{(v_i)}$ is the binary label indicator vector for the image unit located in v_i ; $\tilde{\mathbf{o}}^{(v_i)}$ is the corresponding class likelihood vector, which is upsampled

Figure 4.4: Graphical visualization of the class frequencies (left) and weights (right) on the sift Flow datasets [1]. The classes are sorted in the descending order based on their occurrence frequencies in training images.

from $\tilde{\mathbf{o}}^{(v_i)}$ by the deconvolution network. The errors propagated from DAG-RNN to the pre-trained CNN for image unit v_i are calculated based on the following equations:

$$\Delta x^{(v_i)} = \sum_{\mathcal{G}_d \in \mathcal{G}^U} U_d^T d\mathbf{h}_d^{(v_i)} \circ f'(\mathbf{h}_d^{(v_i)}). \quad (4.7)$$

where the notations share the same meaning with Eqn 4.4, 4.5.

4.5 Attention to Rare Classes

Another important issue in scene segmentation is that the class occurrence distribution is extremely imbalanced. In other words, very few classes take up large percentage of pixels in images. An example is demonstrated in Fig. 4.4, where 6 out of 33 classes fill nearly 85% pixels in images in Sift Flow dataset [1]. This phenomena is shared by other scene segmentation datasets, such as PASCAL Context [11], COCO Stuff [18]. This severely imbalanced data distribution makes the learning problem challenging. On one hand, training examples of infrequent classes are usually far from sufficient for the learner to generalize well, and predictions of rare classes are likely to be smoothed out by frequent classes. On the other hand, we cannot over-exaggerate the importance of rare classes in the learning process, otherwise the visual quality of predicted label maps are full of noisy

and spurious rare-class predictions. Previous researchers have addressed this problem from two perspectives:

- In the patch-based CNN training, Farabet et al. [47] and Shuai et al. [50] oversample the rare-class pixels to augment their training examples. (different sampling methods are introduced and discussed by Shuai et al. in [50].) In the image-to-image structured training, sampling methods can also be applied over losses to achieve the desired class frequency distribution in the training batch, which is essential to address the severe data imbalance in scene segmentation. However, as observed by Long et al. [21], this practice may slow the convergence of network training.
- Mostajabi et al. [48] and Caesar et.al [49] assign an attention weight for each class based on its inverse frequency in the training images. Such class weighting function achieves good performance on less-imbalanced object segmentation, whereas it fails on scene segmentation. The main reason is that the weights for extremely rare classes are exaggeratingly big, so the learning process for frequent classes is significantly suppressed. For example, the frequency ratio between the most frequent (sky) and the most rare class (moon) on the Sift Flow dataset is 3.5×10^4 .

In this chapter, we propose a novel weighting function to modulate the class-wise loss. Specifically, inspired by the “hybrid” sampling in [50] that achieves the best performance tradeoff over all classes, we define the weighting function $\mathbf{w}$ ($\mathbf{w}_j$ is the weight for class j) as follows:

$$\begin{aligned} \mathbf{w}_j &= k^r \\ r &= \max(0, \lceil \log_{10}(\eta/f_j) \rceil) \end{aligned} \tag{4.8}$$

where $\lceil \cdot \rceil$ is the integer ceiling operator, f_j is the occurrence frequency of the class j , η denotes the threshold that discriminates the rare classes. Specifically, a class is identified as rare if its frequency is smaller than η , otherwise, it is a frequent class. k is a re-scaling

factor that modulates the importance of rare classes ($k = 2$ in our experiments). In more details, the *log* operator $m_j = \lceil (\log_{10}(\eta/f_j)) \rceil$ firstly measures the discrete magnitude of rareness for class j (e.g. frequent, rare, extremely rare, etc). The more rare a class appears in the training images, m_j is higher. Then, the k -exponential operator (re-scaling factor) assigns the attention weights based on the rareness magnitude m_j . The max operation is used to keep the attention weights for frequent classes to be unchanged.

The proposed class weighting function achieves a similar functionality as “hybrid” sampling in [50], which has the following properties: (1), it only assigns higher weights for rare classes; (2), the attention weights for rare classes grows exponentially based on their ratio magnitudes w.r.t the threshold η . Thus, rare classes can receive reasonable attention during training by setting the re-scaling factor k properly. Finally, we use the following criterion to determine the value of η : the accumulated frequency of all the non-rare classes is around 85%. We call it 85%-15% rule, and Yang et al. use a similar rule in [94].

4.6 Implementation Details

Fig. 4.1 illustrates the architecture of our segmentation network.

Pre-trained CNN: We use the truncated VGG-16 net [13] (pre-trained on ImageNet [15]) to be the local feature representation module in our experiments. In detail, we discard all the layers after the 5th pooling layer to yield the desired convolutional network. The reasons for not including fully connected layers (**fc6**, **fc7** in VGG-16) are: (1) to reduce network parameters and computations; (2), features output by those two layers are specific to source domain [137], whose knowledge can be hardly transferred.

DAG-RNN: We use a single-layer DAG-RNN to be the context aggregation module. More specifically, non-linear functions in DAG-RNN (refer to Equation 4.2) are ReLU [23] for hidden neurons: $\mathbf{f}(\mathbf{x}) = \max(0, \mathbf{x})$ and **Softmax** for output neurons (g). As suggested

by Le et al. [138], the hidden weight matrices W are initialized with identity matrix. The dimensionality of hidden representations are empirically set to be 256, 256 and 512 for Sift Flow (33 classes), Pascal-Context (59 classes) and COCO-Stuff dataset (171 classes) respectively.³ Besides, the information propagation matrices W in Equation 4.2 are nonidentical for different directional predecessors in each DAG.

Deconvolution network: We use deconvolution network to be the feature map upsampling module. The deconvolution network has three consecutive deconvolution (transposed convolution) kernels [21], which have the upsampling factor of 2, 2, 8 respectively (c.f. Fig. 4.1). Different from [38], our deconvolution network does not include unpooling layers to allow for fast training.

Training details: The proposed segmentation network is trained by batch-based stochastic gradient descent with momentum (batch size = 10). The learning rate is initialized to be 10^{-3} , and it is decayed by 10 times after 15 and 20 epoches (25 epoches in total). Higher ($\times 10$) learning rate is used for DAG-RNN. The momentum is fixed to be 0.9 throughout the learning phase. The images are resized to have maximum length of 448 pixels, and they are zero padded to have square size to allow for batch processing. Besides, we randomly flip the images horizontally (on the fly) to perform image augmentation.

Evaluation details: We forward the images and their horizontally mirrored copies to the trained model (FCN or DAG-RNN), and then we average their corresponding outputs to generate the unary belief maps. We further apply the public code of fully connected CRF [41] to the unary belief maps to generate the final label prediction maps. The parameters of CRF are cross-validated on 500 images on Pascal Context. Similar to other literatures, we use three categories of evaluation metrics:

³We found that further increasing the hidden dimensionality over this threshold does not bring significant performance benefits.

- Global Pixel Accuracy (**GPA**), which is the percentage of all correctly classified pixels.
- Average per-Class Accuracy (**ACA**), which is the mean of class-wise pixel accuracy.
- mean Intersection over Union (**IOU**).

4.7 Main Results

Three challenging scene segmentation datasets are used to compare DAG-RNN with state-of-the-art segmentation methods.

Pascal Context [11] dataset has 10103 images, out of which 4998 images are used for training. The images are from Pascal VOC 2010 datasets, and they are re-labeled as pixel-wise segmentation maps which include 540 semantic classes (including the original 20 classes). Each image has approximately 375×500 pixels. Similar to Mottaghi et al. [11], we only consider the most frequent 59 classes in the dataset for evaluation. Based on the 85%-15% rule, the classes whose frequencies are lower than $\eta = 0.01$ are identified as rare classes. Its quantitative results are listed in Table 4.1.

Sift Flow [1] dataset has 2688 images captured from 8 typical kinds of outdoor scenes. Every image has 256×256 pixels, which belong to one of the 33 semantic classes. We adopt the training/testing split protocol (2488/200 images) provided by [1] to perform our experiments. Following the 85%-15% criterion, the class frequency threshold η is set to 0.05. Statistically, out of 33 classes, 27 of them are regarded as infrequent. The graphical visualization of the weights for different classes are depicted in Fig. 4.4. Its results are demonstrated in Table 4.2.

COCO Stuff [18] dataset is a recently established scene segmentation dataset. It contains 10K images, out of which 9K images are used for training. The images are from Microsoft COCO dataset [44], and the previous un-labeled stuff pixels are further densely

Methods	GPA	ACA	IOU
CFM [139]	n/a	n/a	31.5%
DeepLab [42]	n/a	n/a	37.6%
DeepLab + CRF [42]	n/a	n/a	39.6%
FCN-8s [77]	67.5%	52.3%	39.1%
CRF-RNN [43]	n/a	n/a	39.3%
ParseNet [33]	n/a	n/a	40.4%
ConvPP-8s [140]	n/a	n/a	41.0%
UoA-Context + CRF [30]	71.5%	53.9%	43.3%
DAG-RNN	72.7%	55.3%	42.6%
DAG-RNN + CRF	73.6%	55.8%	43.7%

Table 4.1: Performance comparison on the Pascal Context dataset (59 classes). For fair comparison, we only present methods that use VGG-16 [13] as feature extractor. DAG-RNN (with CRF [41]) achieves the best results.

annotated with extra 91 classes. In total, each pixel belongs to one of 171 semantic classes. The frequency threshold 0.4% is used to determine the rare classes. Quantitative results are shown in Table 4.3.

Results. In comparison with recently developed segmentation networks including FCN[21], DeepLab [42], ConvPP [140] and ParseNet [33], DAG-RNN consistently outperforms them by a noticeable margin on all scene segmentation datasets. It demonstrates that explicit contextual modeling via DAG-RNN is promising as well as essential in scene segmentation. Meanwhile as expected, fully connected CRF [41] further improves the segmentation performance over unary predictions of DAG-RNN, which validates that CRF and DAG-RNN achieve context aggregation via different and complementary ways. As shown in Table 4.1 4.2 4.3, DAG-RNN (with CRF) achieves very competitive results that are comparable or even better than state-of-the-arts on all three datasets. It is important to note that the highly competing method - UoA-Context - concatenates three-scale features (of the image pyramids) in their segmentation network.

Methods	GPA	ACA	IOU
Byeon <i>et al.</i> [141]	70.1%	22.6%	n/a
Liu <i>et al.</i> [1]	74.8%	n/a	n/a
Farabet <i>et al.</i> [47]	78.5%	29.4%	n/a
Pinheiro <i>et al.</i> [56]	77.7%	29.8%	n/a
Tighe <i>et al.</i> [3]	79.2%	39.2%	n/a
Sharma <i>et al.</i> [32]	79.6%	33.6%	n/a
Yang <i>et al.</i> [94]	79.8%	48.7%	n/a
Shuai <i>et al.</i> [50]	81.2%	45.5%	n/a
ParseNet [33]	86.8%	52.0%	40.4%
ConvPP-8s [140]	n/a	n/a	40.7%
FCN-8s[77]	85.9%	53.9%	41.2%
UoA-Context + CRF [30]	88.1%	53.4%	44.9%
DAG-RNN	87.3%	60.2%	44.4%
DAG-RNN + CRF	87.8%	57.8%	44.8%

Table 4.2: Performance comparison on the Sift Flow dataset. DAG-RNN achieves very competitive performance that is on par with state-of-the-arts.

4.8 Detailed study

4.8.1 Comparison with FCN

In this section, we compare FCN with our segmentation network. To this end, we first introduce a baseline FCN, in which the convolution kernel with size $3 \times 3 \times 512 \times 512$ is used to be the context aggregation module. In our segmentation network (c.f. Fig. 4.1), a single-layer DAG-RNN (with hidden dimension of 128) is used to perform context aggregation. As a reference, FCN-8s [21] adopts two layer pre-trained plain CNN (the sizes of convolution kernels are $7 \times 7 \times 512 \times 4096$ and $1 \times 1 \times 4096 \times 4096$). Note that the remaining two modules - pre-trained CNN (truncated VGG-16 [13]) as well as deconvolution network - remain the same across these segmentation networks. Table 4.4 lists their segmentation performance on Pascal Context [11], and Table 4.5 presents the details of their context aggregation modules.

Methods	GPA	ACA	IOU
FCN [18]	52.0%	34.0%	22.7%
DeepLab [42]	57.8%	38.1%	26.9%
FCN-8s [77]	60.4%	38.5%	27.2%
DAG-RNN	62.2%	42.3%	30.4%
DAG-RNN + CRF	63.0%	42.8%	31.2%

Table 4.3: Performance comparison on COCO Stuff dataset (171 classes). DAG-RNN significantly outperforms the popular networks - FCN and DeepLab.

4.8.1.1 Quantitative performance comparison and discussion

In comparison with FCN-8s [21], baseline FCN entails dramatically (2 magnitude) fewer parameters. Interestingly, it even outperforms FCN-8s on Sift Flow, and it slightly lags behind on Pascal Context. This phenomenon can be explained in the following aspect. Convolution kernels with larger spatial size expands FOV more aggressively. As images from Sift Flow are in low resolution (256×256 pixels), baseline FCN has spanned sufficiently large FOV (c.f Table 4.5) so that feature maps encode the context of full image. In this scenario, further expanding FOV is not expected to improve the representative capability of feature maps. On the contrary, larger convolution kernel is more favorable for parsing high-resolution input images. That’s why we observe the performance inconsistency of different FCNs over two datasets with disparate image resolution statistics.

As the context aggregation module is switched to DAG-RNN, we observe a different and interesting story. As clearly shown in Table 4.4, DAG-RNN achieves remarkably better segmentation performance than both FCNs on both datasets. In the first place, these results demonstrate the superiority of DAG-RNN over plain CNN (i.e. conv kernels) to aggregate context. Second, DAG-RNN can potentially handle a broader range of resolutions as the propagation of information across image does not have to be bounded as it is for convolution by kernel size.

4.8.1.2 Architecture comparison

Here, we directly compare the architecture differences over different segmentation networks, which results from context aggregation module. As shown in Table 4.5, DAG-RNN entails significantly reduced parameters comparing to other networks. Besides, DAG-RNN requires significantly less FLoating-point OPerations (FLOPs, floating-point multiplication and addition) to complete a feed-forward pass. Thus, DAG-RNN is more favorable to be potentially applied to resource-constrained embedded devices. Most importantly, DAG-RNN is able to deliver every piece of local context to the rest of images, thus the receptive fields spanned by DAG-RNN are approximately the entire image. In this scenario, local classification is fully contextualized, which largely contributes to the superior performance of DAG-RNN.

We also compare the average inference time of these segmentation networks in practice. We take batch testing (batch size: 10), and we show the average inference frequency (hz) of each segmentation network in Table 4.5. Even though our un-optimized implementation code of DAG-RNN is CPU-based (3.4G HZ, 4 cores), it's still faster than the testing speed FCN-8s [21] on a single GPU (Tesla K40). This is mainly because DAG-RNN demands significantly less computations (i.e. FLOPs) than FCN-8s. Note that the remaining two modules of our segmentation network - pre-trained CNN and deconvolution network - are executed in GPU.

4.8.2 Comparison with Dilated FCN

In this section, we compare DAG-RNN with dilated FCN [34] that represents a very competitive segmentation network architecture. Architecturally speaking, Dilated FCN consists of two disjoint sub-networks: (1), Front-end DilatedNet and (2), Context Network. In detail, DilatedNet is a re-architecture of the pre-trained VGG-16 [13] by removing its two pooling layers (i.e. pool14 and pool15) and applying their following convolutions

Methods	Sift Flow		Pascal Context	
	GPA	IOU	GPA	IOU
FCN-8s[77]	85.9%	41.2%	67.5%	39.1%
Dilated FCN [34]	86.8%	41.5%	71.8%	40.4%
Baseline FCN	86.3%	41.9%	69.5%	38.6%
Baseline FCN+ CRF [41]	87.0%	42.3%	70.1%	39.8%
DAG-RNN	87.0%	44.0%	72.5%	42.3%
DAG-RNN + CRF [41]	87.5%	44.7%	73.4%	43.3%

Table 4.4: Quantitative result comparisons on scene segmentation benchmarks. A typical input image has the resolution of 256 x 256 and 375 x 500 pixels for Sift Flow and Pascal Context datasets respectively. To facilitate study, a light model (hidden dimension of DAG-RNN is 128) is adopted in the controlled experiments. Therefore, there is a slight discrepancy between the performance listed here and the reported ones (DAG-RNN) in Table 4.1, 4.2.

with larger dilation factors. DilatedNet outputs the un-normalized class likelihood feature maps. Context Network is stacked with multiple convolution kernels (with different dilation factors), and it is attached atop DilatedNet to model contextual dependencies over local beliefs. We follow the released model architecture [34] and train it on Pascal Context and Sift Flow datasets. The results are demonstrated in Table 4.4.

Functionally speaking, context network in Dilated FCN [34] is to model contextual dependencies over local beliefs, which behaves like a post-processing module (e.g. CRF). In contrast, DAG-RNN is to model contextual dependencies over image features, and it is an essential module in our segmentation network. As shown in Table 4.4, DAG-RNN consistently and significantly outperforms Dilated FCN on both datasets. It’s important to note that context network simply brings in minor performance benefits over DilatedNet ($< 1.0\%$ mIOU on Pascal Context and Sift Flow).

In Table 4.5, we list the detailed architecture comparison between DAG-RNN and Dilated FCN. It’s important to note that Dilated FCN include two essential context

	# Params	FOV (Pixels)	FLOPs	Inference
Baseline FCN	2.36×10^6	276×276	9.25×10^8	11.5 hz (GPU)
FCN-8s	1.03×10^8	402×402	4.69×10^{10}	9.0 hz (GPU)
Dilated FCN	$>1.1 \times 10^8$	Full Image	$>7.5 \times 10^{11}$	2.8 hz (GPU)
DAG-RNN	2.62×10^5	Full Image	4.50×10^7	10.4 hz (CPU)

Table 4.5: Detailed comparison of Context Aggregation Module (CAM) over different segmentation networks. FLOPs denote the count of floating-point operations. the GPU / **CPU** flag symbolizes that CAM of these comparing networks are executed on the corresponding devices. The batch-based inference testing takes input as 448×448 images (re-sized and zero padded). Feature maps that are fed to CAM have the spatial resolution of 56×56 and 14×14 for DilatedNet and the rest networks respectively.

aggregation modules (CAM) ⁴ : (1), pre-trained **fc** layers in Front-end DilatedNet (CAM-FC, c.f. definition of functional modules in Fig. 4.1.) and (2) context network (CAM-CN). As CAM in Dilated FCN is fed with very dense feature maps, it’s not surprising that it demands significantly heavier computations (FLOPs) to complete their feed-forward computations. As clearly demonstrated in Table 4.5, the inference time of Dilated FCN is significantly longer than DAG-RNN. Meanwhile, Dilated FCN also consumes significantly more parameters than DAG-RNN, which also hinders it to be potentially applied to resource-constrained devices.

4.8.3 Comparison with Fully Connected CRF

In this section, we compare DAG-RNN with fully connected CRF [41][42][43] in terms of contextual modeling. To be more concrete, CRF models the contextual dependencies via minimizing the structured labeling energy which is a combination of negative unary likelihood and clique-based pairwise label compatibility. It is proved that such mechanism is able to greatly smooth the label prediction map as well as to refine its boundary localization. Specifically in our experiments, we adopt our segmentation network (i.e.

⁴Considering that CAM-CN entails significantly less parameters as well as FLOPs comparing to CAM-FC, we present the corresponding statistics bounded by those of CAM-FC in Table 4.5.

Figure 4.5: Detailed class-wise accuracy comparisons on Pascal Context dataset. The classes are presented in the descending order based on their occurrence frequencies, and rare classes are highlighted with **blue** color.

baseline FCN, DAG-RNN) to generate pixel-wise unary belief maps, then we use the same pairwise features as in [42]. Finally, we use the public implementation of CRF[41] to perform the image-wise inference.

4.8.3.1 Quantitative performance comparison

The results are reported in Table 4.4. Overall, DAG-RNN and CRF can both improve the quantitative segmentation performance (over baseline FCN). However, the improvement resulting from DAG-RNN are much more significant than CRF. By looking into the class-wise accuracy changes in Fig. 4.5, we understand that DAG-RNN is able to effectively contextualize many classes so that their parsing accuracies are significantly improved (e.g. dog, cow, sidewalk, etc.). In contrast, CRF slightly improves the performance of many classes without discrimination.

4.8.3.2 Qualitative performance comparison

In order to gain some insights, we present qualitative examples in Fig.4.6. Those examples vividly illustrate how DAG-RNN and CRF improve the visual quality of the prediction maps in different ways. To elaborate,

DAG-RNN is capable of enforcing local consistency. Namely, neighborhood pixels are likely to be predicted with the same labels. Take the example in 1st row and 1st column for illustration, the light reflection in the ‘tv monitor’ makes the semantically identical pixels visually different, so an obvious spurious label prediction blob pops out in the FCN predictions. CRF can substantially smooth out this blob until it reaches the contour of light reflection circle. In contrast, DAG-RNN successfully deals with this situation by smoothing out all the light reflection areas. The same story goes to the image in 1st row and 3rd column, where DAG-RNN parses the neighboring visually different ‘train’ pixels to a consistent region, whereas CRF fails. Simply speaking, CRF behaves in a low-level smoothing manner that simply enforces consistency over visually similar regions. DAG-RNN, on the other hand, enforces high-level semantic neighborhood consistency.

Additionally, DAG-RNN enforces higher-order semantic coherence to a large extent. In other words, pixels that are spatially distant are predicted with labels that usually co-occur in a meaningful scene. This is very essential in scene understanding. The example (1st row, 2nd column) clearly demonstrates that the wrong aeroplane and tree predictions are corrected to bird pixels by DAG-RNN. Another example (2nd row, 4th column) show that the noisy predictions in the upper image are revised to their genuine identities (i.e. ceiling, light). This recovery requires deep understanding to the scene structure as well as its semantics. In contrast, CRF [41] fails to restore their genuine identities.

Nonetheless, DAG-RNN is inferior at localizing object boundaries, so their predicted label maps are less visually pleasing. This is not surprising as low-level appearance cues are essential to accurately localize boundaries. On the other hand, CRF performs very well at localizing boundaries by employing low-level image cues as pairwise features. Thus, they are expected to be complementary to enhance the prediction maps. As shown in Table 4.1, 4.2, 4.3, 4.4, we observe that CRF further boosts the unary predictions of DAG-RNN significantly. Such quantitative performance gains are clearly reflected in the huge improvement to the visual quality of label prediction maps in Fig. 4.6.

4.8.4 Ablation Study

4.8.4.1 Class-weighted loss

The introduced class-weighted loss relates an attention weight to each semantic class based on its occurrence frequency. We can understand that learning rates are modulated for each class specifically, and they are k-exponentially bigger for rare classes. Hence, the rare class pixels are expected to gain higher attention during network learning.

We compare the performance behavior of DAG-RNN trained with and without weighted loss, and their result statistics on Pascal Context dataset are shown in Table 4.6. Not surprisingly, the segmentation network trained without weighted loss prioritizes to frequent classes. As expected, it performs poorly on segmenting out infrequent classes as

Classes	Weighted Loss	GPA	ACA	IOU
All	✓	72.5%	54.7%	42.3%
	×	73.0%	51.5%	41.1%
Frequent	✓	n/a	71.6%	58.8%
	×	n/a	73.3%	59.5%
Rare	✓	n/a	44.7%	32.6%
	×	n/a	38.6%	30.1%

Table 4.6: Quantitative result comparison on Pascal Context dataset (59 classes, including 22 (37) frequent (rare) classes). The introduced class-weighted loss delivers more attention to rare classes during network training. In consequence, the model (trained with weighted loss) performs significantly better at recognizing rare classes, while simultaneously working very well at parsing frequent classes.

well. After the class-weighted loss is adopted, the average parsing accuracies are significantly enhanced for rare classes. Their detailed class-wise parsing accuracy comparison is visualized in Figure 4.5 (second chart). It’s also not hard to observe that many rare classes are parsed more accurately, which validates that infrequent classes are indeed better learned by distributing them higher attention. Meanwhile, parsing performance are marginally suppressed for frequent classes. Overall, such a class-weighted loss brings noticeable improvement to ACA and IOU metrics, while it still maintains competitive GPA. Importantly, the resulting performance gain does not incur any additional computation overhead. Above all, it presents an effective as well as an economical approach to deal with the severely imbalanced class distribution in scene segmentation.

4.8.4.2 Decomposition of UCG

In this section, we instantiate various DAG-RNNs, each of which use different number of DAGs to approximate UCG. Then, we inspect their performance behaviours in Table 4.7. First, we observe a significant performance boost after DAG-RNN (even with only 1 DAG) is engaged in our segmentation network, which clearly demonstrates the usefulness and effectiveness of context aggregation. We also observe that the performance easily

# DAGs	GPA	ACA	IOU
0 DAGs	67.4%	48.8%	36.0%
1 DAGs	71.7%	53.7%	41.2%
2 DAGs	72.4%	54.3%	42.1%
4 DAGs	72.5%	54.7%	42.3%

Table 4.7: Quantitative result comparison of DAG-RNN with different number of DAGs. Note that the segmentation network does not have context aggregation module in the case where no DAGs are used in DAG-RNN.

saturates as we include more than 2 DAGs in DAG-RNN. This result implies that contexts aggregated through different DAG structures are somewhat overlapping and there is no need to use more DAGs. As stated in Section 4.4.3, the decomposition rule requires that any two vertexes be mutually reachable in the set of DAGs $\mathcal{G}^{\mathcal{U}} = \{\mathcal{G}_1, \dots, \mathcal{G}_d, \dots\}$, so that local context can be delivered to any other image regions. Therefore, we use 4 DAGs in our DAG-RNN.

4.9 Discussion

4.9.1 DAG-RNN on PASCAL VOC

We hypothesize that rich and dense (pixel-wise) semantic annotations are essential to fully leverage the effectiveness of DAG-RNN. To elaborate, the features are explicitly connected with graphs in DAG-RNN, and their local information are mutually propagated so that every feature is context aware. However, different from probabilistic graphical models, the message (feature) passing in DAG-RNN is agnostic of semantic (discriminative) class information. To enable such feature passing to be effective towards modeling rich semantic dependencies, rich and dense pixel-wise annotations are necessary to deliver these semantic supervision to DAG-RNN. Take the example in Fig. 4.6 (2nd row and 1st column) for demonstration. In the case where ‘sky’ and ‘tree’ pixels are masked out as ‘background’ in the annotated label maps, it’s hard for any contextual

Figure 4.6: (Best viewed in electronic version with zooming in.) Example of qualitative parsing results on Pascal Context dataset. We present six images (from left to right, top to down) in a block which represent input image, human annotation, label maps predicted by baseline FCN, baseline FCN + CRF [41], DAG-RNN and DAG-RNN + CRF respectively. The examples in the first row demonstrate the case in which recognition accuracies of “thing” (i.e. tv monitor, bird, train and boat) are phenomenally increased after DAG-RNN is used. The second-row examples show that DAG-RNN improves the parsing performance for “stuff” (i.e. floor, water, wall, road, etc). The third-row examples present some interesting cases, where wrong predictions (such as ‘ground’ or ‘dog’) also make sense.

modeling mechanism to capture the rich semantic dependencies that have been lost in the supervision signal (annotated labels), as ‘aeroplane’, ‘bird’ or any other objects can coexist with ‘background’.

To test how DAG-RNN works on sparsely annotated data, we apply our segmentation network on Pascal VOC 2012 dataset, where most image pixels are simply annotated as ‘background’. The testing mean IOU of DAG-RNN is 69.4% (with CRF: 70.4%), which noticeably outperforms FCN-8s [21] (67.5%) as well. However, such result is not competitive when it is positioned with DeepLab [42] (71.6%) Dilated FCN (with COCO pre-training: 73.5%) and UOA-Context [30] (75.3%), even though DAG-RNN outperforms them consistently on three scene segmentation datasets. As clearly demonstrated, DAG-RNN can still work for object segmentation, but it is more suitable for scene segmentation where contextual dependencies are richer and more demanding.

4.9.2 Context vanishing in DAG-RNN

In a similar spirit to vanishing gradient problem [142], local contexts may vanish before they are propagated to spatially distant features. In this scenario, higher-order contextual dependencies fail to be captured by DAG-RNN. To address this issue, we have introduced the UCG with 8-neighborhood system. As shown in Fig. 4.7, the diagonal connections introduce a significant shorter propagation paths, which makes the context vanishing problem less severe. In our experiments, we do observe promising performance improvement (0.5% mIOU, on Pascal Context [11]) by adopting UCG(8) than by UCG(4). For example, a typical feature tensor (c.f. Fig. 4.1) in our experiment has the dimensionality of $14 \times 14 \times 512$, so the shortest propagation paths in DAG(8) have a length of only 13 steps. As demonstrated in [138], the proper nonlinearity as well as the identity weight initialization enable the context should not be vanished under such DAG(8).

Figure 4.7: Two UCGs (with 4, 8 neighborhood system) and their induced DAGs in the northwestern (NW) direction. In comparison with DAG(4), DAG(8) enables information to be propagated in shorter paths, which is critical to prevent the long-range information from vanishing. As an example, the length of propagation path from v_9 to v_1 in $\mathcal{G}_{nw}^8$ is halved to that in $\mathcal{G}_{nw}^4$ ($4 \rightarrow 2$ steps) due to the shortcut path.

4.10 Conclusion

In this chapter, we discuss the challenges of scene segmentation. To address these challenges, we firstly propose DAG-RNN to capture the rich contextual dependencies over image region features. More specifically, we place DAG-RNN on top of a pre-trained CNN to give rise to the backbone of our segmentation network. Furthermore, we propose a novel class weighted loss to train our segmentation network, which is used to distribute reasonably more attention for rare classes.

We have extensively compare DAG-RNN with FCN and fully connected CRF, and we show that DAG-RNN is more effective in terms of context aggregation. Concretely, DAG-RNN demonstrates significantly better segmentation performance than its FCN counter-

parts. Besides, it entails dramatically fewer parameters and computations, so DAG-RNN is more favorable to be potentially applied to resource-constrained embedded devices. In comparison to CRF, DAG-RNN is able to enforce high-level semantic consistency, which contributes to its quantitative performance superiority to a large extent. Meanwhile, the class-weighted loss is experimentally proved to be effective at significantly improving the parsing performance of rare classes. Importantly, DAG-RNN (together with CRF) has achieved state-of-the-arts on three challenging scene segmentation datasets.

Chapter 5

IFCN: towards achieving robust high-level and low-level scene parsing

5.1 Introduction

Scene segmentation refers to parsing a scene image into a set of coherent semantic regions. It demands multi-level parsing ranging from low-level (e.g. boundary localization) to high-level (e.g. object recognition and localization). Thus, it's understandable that a well-performed segmentation network should be able to effectively incorporate different scale information.

State-of-the-art segmentation networks are based on pre-trained classification network (e.g. CNN). However, detailed spatial information are largely lost in its output feature maps. In order to incorporate the detailed image features, researchers usually adapt the architecture of pre-trained CNN [22][21][34] based on two approaches: (1), “dilation” - it removes subsampling layers (`pool` or strided `conv` layers with stride greater than 1) in pre-trained CNN and then performs subsequent convolution operations with higher dilation factors (i.e. stride rate); or (2), “skip” - it adds skip branches (usually linear classifier) from early layers of pre-trained CNN such that low-level features are explicitly incorporated into image parsing. Recent literatures [22][34][40] as well as our experiments

Figure 5.1: (Best viewed in color) Semantic segmentation demands robust high-level as well as low-level parsing. IFCN outperforms FCN both in high-level smoothing / recognition (“train” and “ground” in the first and third example) and low-level boundary localization (e.g. bird legs in second example).

suggest that “dilation” network¹ outperforms its “skip” counterpart. In this chapter, we demonstrate that the parsing performance of “skip” network can be significantly improved by re-parameterizing the skip layers. Next, we introduce “dense skip” architecture to retain rich set of low-level information from pre-trained CNN. In detail, we consider every possible informative lower-level feature maps into semantic parsing. These dense skip layers enable the segmentation network to incorporate very rich low-level contexts, which is essential to improve its performance on low-level parsing. In comparison with the popular “dilation” network, we demonstrate that “dense skip” architecture is more effective as well as more efficient in terms of retaining low-level detailed information from pre-trained CNN.

¹Hereafter, we simply use “skip” and “dilation” network to notate the above two network architectures.

Feature maps generated by pre-trained CNN are usually robust enough for image-level visual recognition, but they are not equally discriminative and representative at every spatial location. Take images from Figure 5.1 for example, local features for upper-right “train” regions (first image) and lower-right “ground” regions (third image) are not discriminative such that their unary predictions are noisy. Thus, feature context aggregation (CA) is essential to improve their representative capability. In this chapter, we propose and place a convolutional context network (CCN) on top of pre-trained CNN to aggregate context for high-level feature maps. It is empirically demonstrated that CCN is able to significantly improve the parsing performance of segmentation networks. Meanwhile, CCN delivers a more effective CA module than many state-of-the-art [22][33][76][34] including Atrous Spatial Pyramid Pooling (ASPP) in DeepLab [22] and DAG-RNN [76].

Our segmentation network is fully convolutional, and it has dense skip layers from pre-trained CNN and CCN. Thus, we name it improved FCN (IFCN) based on its enhanced structure over FCN. The qualitative parsing results in Figure 5.1 clearly illustrate that IFCN outperforms FCN both in terms of high-level as well as low-level parsing. Overall, IFCN presents a strong segmentation network architecture, which achieves state-of-the-art performance on standard semantic segmentation datasets including ADE20K [12], Pascal Context [11], SUN RGB-D [143] and Pascal VOC 2012 [20]. In summary, this chapter makes the following contributions to yield the network architecture of IFCN.

- We discuss and compare two adaptation approaches to retain detailed spatial information from pre-trained CNN - “dilation” and “skip”. By slightly modifying the parameterization of skip layers, we improve the parsing performance of “skip” network significantly.
- We further propose “dense skip” architecture to retain rich set of low-level information from pre-trained CNN, which is demonstrated to be more effective as well as more efficient than its “dilation” counterpart.

- We propose a convolutional context network (CCN) to aggregate context for high-level feature maps. We show that CCN is a very effective to boost the parsing performance of segmentation networks.

We also compare qualitative results of IFCN with state-of-the-art segmentation networks, which gives deeper insight to understanding our contributions .

5.2 Segmentation Networks

Scene segmentation requires both robust low-level and high-level parsing. Specifically, low-level spatial information is important to detect small-size objects as well as to delineate their boundaries. On the other hand, context-aware and high-level feature representation is essential to recognize stuff regions (e.g. ‘building’ and ‘ground’ in Figure 5.1) and appearance-inconsistent or zoom-in objects (e.g. ‘train’ in Figure 5.1). In this chapter, we discuss how to integrate robust high-level and low-level parsing in an efficient segmentation network.

5.2.1 Dilation or Skip?

Pre-trained CNN (i.e. local representation module) usually outputs high-level semantic features that are essential for robust object recognition. However, low-level and mid-level features are equally important for image parsing as well. For example, localizing small-size objects as well as delineating object boundaries requires robust low-level visual recognition. Thus, how to retain and incorporate the detailed spatial information from pre-trained CNN remains a promising research direction. Overall, two prominent approaches are developed to adapt pre-trained CNN in order to either mitigate the significant loss of detailed spatial information or explicitly incorporate lower-level feature maps into semantic predictions.

- **Dilation.** This line of work consists of representative segmentation networks such as DeepLab [22] and DilatedNet [34]. In detail, they propose to remove several

Networks	Type	IOU	Speed (hz)
FCN-8s (Orig) [12]	skip	29.5%	10.0
FCN-8s	skip	31.4%	10.0
FCN	dilation	32.8%	3.60
FCN-8s	our skip	32.6%	10.0
DeepLab-v2	skip	33.5%	10.8
DeepLab-v2	dilation	35.1%	7.30
DeepLab-v2	our skip	34.7%	10.8

Table 5.1: Two approaches - “Dilation” and “Skip” - are used to adapt pre-trained CNN (in this case, VGG-16 [13]). Results of their corresponding networks are evaluated on ADE20K dataset [12]. Networks are trained under the same condition (details are elaborated in Section 5.3) except that FCN-8s(Orig) is trained without class-weighted loss [76]. Speed refers to the inference time of segmentation networks on a single NVIDIA Titan X.

submapping layers (i.e. **pool** layers, **conv** layers with strides larger than 1), and modify subsequent **conv** kernels with dilated (atrous) **conv**. By doing this, resolution of feature maps does not reduce significantly, and the detailed scene layout is still maintained to a large extent. In this regard, the detailed spatial information is not significantly lost in the output feature map especially comparing with the original one.

- **Skip.** The seminal work FCN [21] adopts this architecture design. Specifically, skip layers are used to fuse predictions from early feature maps of pre-trained CNN. Thus, lower-level features are explicitly incorporated in semantic segmentation.

We experimentally compare and discuss these two network adaptation methods on ADE20K [12]. Results are summarized in Table 5.1. Consistent with current findings in [22][34], our experiments show that “dilation” network demonstrates superior parsing performance than its “skip” counterpart. However, the inference speed of “dilation” networks is significantly slower, as they need to process denser feature maps. This undesirable property drives us to explore how to better retain detailed lower-level features under the efficient “skip” architecture.

Figure 5.2: Skip layers in segmentation networks. The right figure depict its detailed computational blocks. Mathematically, $\mathbb{H} = \mathbb{H}' + S(\mathbb{F}, \Theta)$, where $\mathbb{F}$ and $\mathbb{H}$ denotes input feature map (with the dimensionality of *height* $\times$ *width* $\times$ *#channels*) and its class-likelihood map respectively, $S(\cdot, \cdot)$ is the computational function of skip layer and Θ is their learnable parameters. $\mathbb{H}'$ represents class-likelihood maps generated from other branches.

We assume that skip layers are less well trained, which largely contributes to the inferior results of “skip” networks. Referring to Figure 5.2, the gradients that back propagated to pre-trained CNN are $\Delta\mathbb{F} = \frac{\partial\mathcal{L}}{\partial\mathbb{H}} \frac{\partial\mathbb{H}}{\partial\mathbb{F}}$, where $\mathcal{L}$ refers to the class-weighted loss [76]. Consider the usual parameterization of skip layer (1×1 conv kernel - parameterized by W), $\Delta\mathbb{F} = W^T \frac{\partial\mathcal{L}}{\partial\mathbb{H}}$, thus strong error signals $\frac{\partial\mathcal{L}}{\partial\mathbb{H}}$ are not functionally modulated before they are propagated to early layers of pre-trained CNN. In this case, gradient norm $\|\Delta\mathbb{F}\|$ is usually large, which fluctuates the network training. In practice, magnitude smaller learning rates are usually used for skip layers [77] to prevent the network training from divergence. To address this issue, we use a 2-layer convolutional network with batch normalization (BN) [144] to parameterize skip layers (i.e. **Conv+BN+ReLU+Conv**). Specifically, batch normalization is used to stabilize the back-propagated error signals $\|\Delta\mathbb{F}\|$. Thus, skip layers can be better trained by using normal (or larger) learning rate without posing optimization issues.

We adapt two well-known segmentation networks (FCN [77] and DeepLab-v2 [22]²) based on the above “dilation” and “skip” approaches. Their results are shown in Table 5.1. Performance of the same “skip” networks are significantly boosted ($> 1\%$ IOU) after our new parameterized skip layers are used. It’s important to note that such performance benefit vanishes as we apply magnitude smaller learning rate (as used in original skip layers) to our new skip layers, which demonstrates that the performance gain mainly originates from better trained skip layers. As our parameterization of skip layers bring negligible extra computations, inference of “our skip” networks remains efficient. Importantly, “our skip” networks achieve comparable parsing performance with its “dilation” counterpart.

5.2.2 Dense skip: a more effective approach to retain low-level information

In order to retain rich set of low-level information from pre-trained CNN, we propose “dense skip” architecture. In this sense, the corresponding network is expected to perform better on low-level parsing. Besides, it is able to incorporate very rich-scale context to make semantic predictions, which is essential to address the huge scale variance of objects in scene segmentation as well. Take VGG-16 [13] as an illustration, “dense skip” network adds skip layers³ for each intermediate feature map after `pool3` (i.e. feature maps whose stride is larger than 8). Accordingly, we adapt two segmentation networks (FCN [77] and DeepLab-v2 [22]) to “dense skip” architecture. Note that original FCN is a “sparse skip” network. We compare their performance in Table 5.2. As expected, “dense skip” architecture outperforms the usual “sparse skip” counterpart by a noticeable margin, and

²The architecture of DeepLab-v2 [22] is illustrated in Figure 5.5. In order to adapt its “dilation” architecture to “skip”, we revise the dilation factors (stride rate) in the four branches of ASPP to $\{1, 3, 4, 6\}$ respectively. Their corresponding dilation factors in “dilation” network are $\{4, 12, 16, 24\}$, which are close to their original settings $\{6, 12, 18, 24\}$ in [22].

³The fusion strategy for “dense skip” is exactly the same with that of “sparse skip” architecture in FCN [21]. A detailed graphical demonstration of multi-scale fusion in “dense skip” architecture is depicted in Figure. 5.4

Networks	Type	IOU	Speed
FCN	sparse skip	32.6%	10.0
FCN	dense skip	33.6%	9.10
DeepLab-v2	sparse skip	34.7%	10.8
DeepLab-v2	dense skip	35.4%	10.1

Table 5.2: Parsing performance comparison between “sparse skip” and “dense skip” network architectures on ADE20K dataset [12]. Here skip refers to our new parameterized skip layers.

its inference speed remains competitively fast. In comparison with “dilation” network (c.f. Table 5.1), “dense skip” architecture also offers a more effective as well as more efficient framework to preserve low-level information.

5.2.3 Context Aggregation for high-level features

Feature maps generated by pre-trained CNN generally encode high-level semantic information, which are globally discriminative for object/scene recognition [13][19]. However, they are not equally discriminative at every spatial location especially for zoom-in objects and “stuff” regions. Take the first example in Figure 5.1 as illustration, the “train” pixels are visually inconsistent, and they can hardly be recognised unless context from distant “railway” regions are incorporated into semantic parsing. Thus, incorporating context into feature maps is of great significance to achieve the desired robust high-level parsing.

Suppose pre-trained CNN outputs a feature tensor $\mathbb{F}$. Take $\mathbb{F}$ as input, context aggregation module (CA) is to output $\mathbb{F}'$, in which each feature is well contextualized. Our assumption for a good contextualization mechanism is that features in $\mathbb{F}'$ should engage all elements in $\mathbb{F}$. In this chapter, we propose to use basic convolutional kernels to achieve this goal. Different from atrous convolution [22][34], there will be no “holes” during the convolution operations. Thus, it can incorporate every feature within the contextual window into corresponding features in $\mathbb{F}'$. In contrast, dilated (atrous) convolution can

Figure 5.3: **convolutional context network (CCN)** is a convolutional network with dense skip layers. Contextual field (orange coverage in $\mathbb{F}'$) of feature maps are progressively expanded. Note that features in $\mathbb{F}'$ incorporates all elements in $\mathbb{F}$.

only incorporate a fraction of features within contextual windows. This issue is also pointed out by [40] as “gridding issue”.

Convolutional Context Network (CCN). We present its architecture in Figure 5.3. As shown, several **conv** blocks are chained to progressively expand the contextual view of feature maps. Dense skip layers are also used in CCN to aggregate multi-scale contexts.

Relation with ASPP in DeepLab [22]. ASPP aggregates multi-scale contexts by combing multiple **CA** branches, each of which uses dilated **conv** kernels with different stride rates to incorporate different scale contexts. Functionally speaking, dense skip layers in CCN offer a similar way to aggregate multi-scale context. However, receptive field of feature maps $\mathbb{F}'$ in CCN are progressively expanded, so features in $\mathbb{F}'$ engage information from all the elements within the receptive field in $\mathbb{F}$. In contrast, even though feature maps $\mathbb{F}'$ in some ASPP branches can expand a large contextual field (due to large stride rate), features in $\mathbb{F}'$ still incorporates information from very few elements in $\mathbb{F}$.

Relation with DAG-RNN [76]. DAG-RNN propagates local features to different regions of the image. Thus, features in $\mathbb{F}'$ are able to incorporate information from all elements in $\mathbb{F}$. By analysing the unrolled structure of DAG-RNN, we discover that there are skip branches that connect every intermediate feature map with output layers. In this

sense, each intermediate features receive direct supervision signals, which explains why such a deep unrolled network can be effectively trained. Dense skip layers in CCN offer a similar functionality to ease the network optimization difficulty. However, considering that the unrolled DAG-RNN is extremely deep (more than hundred layers), long-range context may vanish[142] during propagation, which limits its capability of contextual modeling. This issue is less likely to happen in the relatively shallower CCN.

Relation with CRF [41]. CRF is usually applied to class likelihood maps to contextualize local beliefs. In detail, the pairwise energy term is largely based on low-level image cues (e.g. color, gradient, position, etc), thus it enforces low-level consistency such as sharpening boundary, appearance-based smoothing, etc. Similar to ASPP [22] and DAG-RNN [76], our CCN is to contextualize high-level features so that high-level consistency can be achieved. In this sense, CRF can be placed on top of our segmentation network to further boost its parsing performance.

We carefully examine and compare these CA modules in a controlled experiment. For efficiency reasons, we adapt the segmentation networks based on “sparse skip” architecture. Their parsing performance on ADE20K are summarized in Table 5.3. Here, CCN is a 8-layer convolutional network (3×3 kernels with 1024 channels) with dense shortcut branches. As expected, segmentation networks with CA modules outperforms baseline FCN, which demonstrates that context aggregation is essential to enhance feature representation. Meanwhile, CCN outperforms CRF, DAG-RNN and ASPP by a significant margin, which clearly justifies our contributions as well as demonstrates its architecture superiority of feature context aggregation.

Comparison with context network in DilatedNet [34]. Architecturally speaking, CCN has dense skip layers to aggregate multi-scale contexts, whereas context network in [34] is a plain CNN. Functionally speaking, context network in [34] is placed on top of classification layer (c.f. Figure 5.5), thus it models the contextual dependencies over class-likelihoods to refine local beliefs. In contrast, CCN is fed with high-level feature

Networks	CA module	IOU
Baseline FCN-8s	NIL	29.5%
Baseline FCN-8s + CRF	CRF	31.1%
FCN-8s[77]	fc6&fc7	32.6%
DAG-RNN [76]	DAG-RNN	33.5%
DeepLab-v2 [22]	ASPP	34.7%
Ours	CCN	36.4%

Table 5.3: Segmentation networks are adapted to “our skip” architecture, and they differentiate each other in terms of context aggregation. Results are evaluated on ADE20K dataset.

maps, and it aggregates and distills context to local features such that their representative capability is improved. It is an essential component in segmentation networks to enhance high-level parsing consistency. By comparing their qualitative results in Figure 5.6, we clearly see that CCN noticeably outperforms context network in terms of enforcing high-level consistency.

5.2.4 IFCN

Our segmentation network is fully convolutional, and it has dense skip layers from both pre-trained CNN and CCN. Thus, we name it improved FCN (IFCN) based on its enhanced structure over FCN. First, “dense skip” architecture is used to retain and incorporate low-level information from pre-trained CNN, which is expected to enhance low-level parsing. Besides, CCN is introduced to aggregate context for high-level feature maps, which brings benefits to high-level visual parsing. Examples in Figure 5.1 qualitatively illustrate such benefits to the resulting parsing maps. It’s important to mention that our contributions in this chapter are orthogonal to many recent techniques that advance segmentation performance. For example, Zhao et al. [46] explores to leverage global scene information to improve parsing performance. Wu et al. [45] exploits to improve the network architecture of pre-trained CNN. Wang et al. [40] proposes to replace bilinear upsampling with convolution to preserve detailed information.

Figure 5.4: Network architecture of IFCN-xs. Similar to the demonstration of FCN-xs, feature maps are shown as grids that reveal relative spatial coarseness. IFCN-4s and IFCN-2s can be trivially inferred from the above architecture demonstration. The black vertical line denotes the conv layer in the pre-trained CNN (VGG-16). In order to save space, some feature maps are not displayed. Note that the detailed architecture of convolutional context network (CCN) can be retrieved in Figure 5.3.

In this chapter, we follow the definition of [76] to illustrate the architecture of segmentation networks. Then, we compare the architecture of IFCN with state-of-the-art segmentation networks in Figure 5.5. These networks differentiate each other from two aspects: (1), how to adapt pre-trained CNN (“sparse skip”, “dense skip” or “dilation”) to retain detailed spatial information; (2), how to aggregate context for high-level feature maps. In Table 5.4, we list their parsing performance on ADE20K dataset [12]. IFCN outperforms all other networks by a significant margin, which clearly justifies our contributions.

5.3 Implementation Details

Network setups. Following the functional modules of segmentation networks defined in [76] (c.f. Figure 5.5), we present the detailed network architecture of our IFCN.

Figure 5.5: Network architecture comparison. These networks differentiate each other from two aspects: (1), how to adapt pre-trained CNN to retain detailed spatial information; (2), how to aggregate context for high-level feature maps.

- **Local Representation.** We use truncated VGG-16 [13] (pre-trained on ImageNet [14]) as our local representation module. In detail, layers after `pool5` are discarded. Given an input image with size 512×512 , it will output feature tensor $\mathbb{F}$ with dimensionality of $16 \times 16 \times 512$. Besides, we adapt pre-trained CNN based on the proposed “dense skip” architecture to retain detailed low-level information.
- **Context Aggregation.** Our CCN is an 8-layer 3×3 (or 6-layer 5×5) convolutional network with dense skip layers (c.f. Figure 5.3), and the hidden dimension is set to 1024 (or 512). Thus, feature in $\mathbb{F}'$ is able to incorporate context from entire image.
- **Upsampling.** We use convolution transpose (deconvolution) kernels [21] to perform upsampling operation.

Class-weighted loss. To distribute more attention for infrequent classes, we modulate the pixel-wise loss according to its rareness magnitude as in [76]. This practice is economic and it is essential to significantly boost the parsing performance of rare classes. As demonstrated in Table 5.1, a significant 2% IOU improvement is observed on ADE20K dataset when the class-weighted loss is used to train FCN. We follow the 85% -15% rule to determine the rare categories, and we refer the readers to [76] for detailed description.

Networks	GPA	ACA	IOU	Speed (hz)
FCN(Orig) [12]	71.3%	40.3%	29.4%	10.0
FCN [77]	73.6%	44.8%	32.6%	10.0
DAG-RNN [76]	73.9%	49.0%	33.5%	9.80
DilatedNet [34]	74.1%	47.9%	33.6%	3.50
DeepLab-v2 [22]	75.1%	48.4%	35.1%	7.30
IFCN (sparse skip)	75.8%	50.4%	36.4%	9.80
IFCN (dense skip)	76.2%	51.7%	37.3%	9.00

Table 5.4: Parsing performance of different networks (c.f. Fig 5.5) on ADE20K dataset.

Training details. Networks are trained with SGD with momentum (batch size 10). The learning rate is initialized to be 10^{-3} , and it is decreased by 10 times after 15 and 20 epoches (25 epoches in total). The momentum is fixed to 0.9. New parameters engaged in CCN and skip layers are randomly initialized (Gaussian distribution with variance 10^{-2}). Meanwhile, higher learning rate ($10\times$) is used for newly-initialized parameters. Images are resized to have maximum length of 512 pixels, and they are zero padded to 512×512 pixels to allow for batch processing. We randomly flip the images horizontally (on the fly) to augment the training images. It is important to note that all the segmentation networks (in the controlled experiments) are trained with exactly the same settings.

Evaluation. Three performance metrics are used to evaluate our IFCN: Global Pixel Accuracy (**GPA**), Average Class Accuracy (**ACA**) and **IOU**.

5.4 Architecture Evaluation on ADE20K

ADE20K [12] is a recently established large-scale dataset for ImageNet scene parsing challenge. The dataset contains 20210 training, 2000 validation and 3352 test images. Each pixel is annotated with one of 150 semantic categories including object classes as well as stuff classes. In order to modulate the rareness weighted loss [76], those classes whose frequency is less than 1% are considered as rare. We report the results on validation images.

Networks	GPA	ACA	IOU
IFCN-32s	73.7%	48.4%	33.2%
IFCN-16s	75.4%	51.5%	36.3%
IFCN-8s	76.0%	52.0%	37.0%
IFCN-4s	75.8%	51.3%	36.0%

Table 5.5: Performance comparisons of different IFCN architectures on ADE20K dataset. For efficiency reasons, the hidden dimension of CCN is reduced to 512, thus a slight performance discrepancy is observed when it is compared with that in Table 5.4.

5.4.1 Which skip layers should be included?

It’s important to note that skip layers from CCN (c.f. Figure 5.3) are essential to aggregate multi-scale context as well as to ease network optimization difficulties (c.f. Section 5.3), and removal of them dramatically deteriorates ($> 1\%$ IOU) its parsing performance. Thus, dense skip layers from CCN should be kept in IFCN. Here, we only discuss which skip layers should be included from pre-trained CNN. To this end, we progressively add skip layers from pre-trained CNN and monitor the performance of their corresponding networks. For efficiency reasons, we consider adding a block of skip layers per evaluation.⁴ In a similar notation to FCN-xs [77], we notate these networks as IFCN-xs. Their results are shown in Table 5.5, where we can see that IFCN-8s performs the best among all “dense skip” architectures. Though IFCN-4s entails more skip layers, it achieves inferior segmentation results. We conjecture that the added block of skip layers in IFCN-4s (comparing to IFCN-8s) incorporates very low-level and noisy feature maps that largely contributes to the degraded performance.

5.4.2 Can the proposed architecture work for better pre-trained CNN?

We have shown that IFCN significantly outperforms FCN and state-of-the-art VGG-16 based segmentation networks in Table 5.4, which clearly justifies our contributions of

⁴ Feature maps are considered to be in the same block as long as they have the same spatial resolution. Take VGG-16 [13] as an example, feature maps `pool4`, `conv5_1`, `conv5_2`, `conv5_3` have the same resolution, so skip layers originating from them are deemed as in the same block.

Networks		ResNet-50	
FCN-8s	74.4%	47.4%	33.9%
IFCN-8s	77.0%	53.2%	38.1%
Networks		ResNet-101	
FCN-8s	76.0%	50.1%	35.8%
IFCN-8s	77.7%	55.4%	39.7%

Table 5.6: Performance (GPA, ACA, IOU) of different network architectures (with different pre-trained CNNs) on ADE20K dataset.

architecture design. In this section, we examine whether such architecture is universally beneficial to different pre-trained CNNs especially more powerful ResNets [19]. To this end, we adapt the pre-trained ResNets (ResNet-50 [19] and ResNet-101 [19]) to FCN-8s and IFCN-8s correspondingly. Their parsing performance on ADE20K are listed in Table 5.6, where we can see that IFCN consistently and significantly outperforms its FCN counterpart, and such performance benefits ($\sim 4\%$ IOU) does not vanish when the proposed architecture design is applied to better pre-trained CNN. This encouraging result clearly suggests that our contributions of this chapter are orthogonal to the general research that improves pre-trained CNN architecture.

5.5 Results on segmentation benchmarks

5.5.1 Evaluation datasets

ADE20K results are listed in Table 5.4 ,5.6.

Pascal Context results are shown in Table 5.7. Pascal Context [11] has 10103 images, out of which 4998 images are used for training. The images are from Pascal VOC 2010 datasets, and they are re-labeled as pixel-wise segmentation maps which include 540 semantic classes (including the original 20 classes). Similar to Mottaghi et al. [11], we only consider the most frequent 59 classes in the dataset for evaluation. Based on the rareness identification rule in [76], those classes whose frequencies are lower than 1% are identified as rare.

Methods	GPA	ACA	IOU
FCN-8s [77]	67.0%	50.7%	37.8%
DeepLab-v1 [22]	n/a	n/a	37.6%
CRF-RNN [43]	n/a	n/a	39.3%
ParseNet [33]	n/a	n/a	40.4%
UoA-Context + CRF [30]	71.5%	53.9%	43.3%
DAG-RNN + CRF [76]	73.6%	55.8%	43.7%
IFCN	74.5%	57.7%	45.0%

Table 5.7: **Pascal Context** validation accuracies. For fair comparisons, we only include methods that use VGG-16 as base network.

Methods	GPA	ACA	IOU
FCN [21]	68.18	38.41	27.39
DeconvNet [38]	66.13	33.28	22.57
SegNet [37]	72.63	44.76	31.84
DeepLab [22]	71.90	42.21	32.08
IFCN	76.90	53.46	40.74

Table 5.8: **SUN-RGBD** (37 classes) testing accuracies (%). We only use RGB modality in our experiments.

SUN-RGBD Results are reported in Table 5.8. SUN-RGBD [143] contains images from NYU depth V2 [145], Berkeley B3DO [146], SUN3D [147] as well as the newly captured images. It has 10335 images in total, out of which 5285 images are used for training. The rareness frequency threshold is fixed to 2.5% based on the 85%-15% rule [76]. We follow previous literatures [37] to consider 37 classes for evaluation. Note that we only use RGB modality as input.

Pascal VOC 2012 results are presented in Table 5.9. Pascal VOC 2012 originally contains 2913 train and validation images, 1456 testing images. Images have approximately 375×500 pixels, each of which belongs to one of the pre-defined object or background categories. We follow [21] to augment the training set from [148]. Thus, we end up with having 12031 training images. The rareness frequency threshold of is set to 1.5%. We submit our parsing maps of testing images to the evaluation servers to report our results.

Figure 5.6: (Best viewed in color with 300% zooming-in) Qualitative comparisons of parsing maps. DAG-RNN [76] is very good at enforcing high-level consistency (first three examples). Due to the adopted “dilation” architecture, DeepLab-v2 [22] and DilatedNet [34] perform well at localizing boundaries (4th and 5th example). IFCN performs equally well in low-level as well as high-level parsing.

Methods	IOU
FCN-8s [77]	67.5%
ParseNet [33]	69.8%
DeepLab-v1 + CRF [42]	71.6%
CRF-RNN [43]	72.0%
DilatedNet [34] (COCO pre-trained)	73.5%
DPN [29]	74.1%
UoA-Context + CRF[30]	75.3%
LRR [149]	74.7%
IFCN	75.2%

Table 5.9: **Pascal VOC 2012** testing accuracies. For fair comparisons, we only include methods that use VGG-16 as base network and does not use auxiliary segmentation masks for pre-training.

5.5.2 Result Summary

IFCN demonstrates significantly better quantitative parsing performance than FCN on all segmentation benchmarks. As the evaluation datasets cover wide range of scenarios that include object segmentation [20], outdoor scene parsing [12][11] as well as indoor scene labeling [12], it’s fair to conclude that IFCN is a versatilely better segmentation network architecture than FCN. By comparing their qualitative parsing results in Figure 5.1, 5.6, we observe that IFCN outperforms FCN both in terms of high-level as well as low-level semantic parsing, which clearly aligns with our motivations. The significant enhanced quantitative performance as well as noticeable improved visual quality of parsing maps clearly justifies our contributions of this chapter.

IFCN vs State-of-the-art. IFCN also outperforms other recently developed parsing networks including DilatedNet [34], DAG-RNN [76], DeepLab [22], etc. By comparing their parsing maps in Figure 5.6, it is easy to find that IFCN achieves the best parsing performance both in low-level and high-level parsing.

5.6 Conclusion and Discussion

In this chapter, we explore how to develop effective as well as efficient segmentation networks that can perform both well at high-level and low-level semantic parsing. To this end, we first discuss and compare two popular adaptation approaches to pre-trained CNN -“dilation” and “skip”. By slightly modifying the parameterization of skip layers, “our skip” network outperforms the usual “skip” network by a noticeable margin. Furthermore, we propose “dense skip” architecture to efficiently retain rich set of low-level information. In addition, we propose a convolutional context network (CCN) to aggregate context for high-level feature maps so that their representative capability can be improved. The resulting network architecture (IFCN) performs competitively well on standard scene segmentation datasets. Without bells and whistles, IFCN achieves state-of-the-art on ADE20K, Pascal Context, SUN RGB-D and Pascal VOC 2012 datasets.

Dilation and Skip? Without considering efficiency issues, it’s technically possible to integrate “dilation” with “dense skip” architecture to a unified network so that lower-level spatial information can be retained from different and possibly complementary manners. We leave this architecture exploration for future work.

Chapter 6

Conclusion and future work

6.1 Conclusion

In this thesis, we start with introducing the challenges inherent to scene segmentation, which is disparate from well-known object segmentation tasks. We figure out that high-level contextual reasoning is significantly more demanding in scene segmentation. In addition, class frequency distribution is significantly more imbalanced in scene segmentation, which needs to be appropriately addressed. Finally, images in scene segmentation usually include more small-size objects, fragmented stuff regions as well as more complicated region boundaries, which requires carefully preserve lower-level image cues.

In order to address these challenges, we present three segmentation models. In a nutshell, the first segmentation model is based on Patch-Convolutional Neural Network (Patch-CNN) ¹, and its labeling results are further refined by integrating cues from global context. The second and third segmentation models are based on more advanced feature extractor (pre-trained CNN, such as VGG-16 [13] or ResNet [19]), and more importantly they are end-to-end trainable. Concretely, we propose two high-level context aggregation layers – DAG-RNN and Convolutional Context Network (CNN), and place them on top of pre-trained CNN to yield the corresponding segmentation network. In the mean time, we propose a novel class-weighted loss to train these two segmentation networks,

¹A convolutional neural network is trained from image patches in the target scene segmentation dataset.

which significantly improve their accuracy at segmenting rare classes. In align with the challenges mentioned earlier, we elaborate and discuss the contributions of these three segmentation models as follows:

- **Semantic contextual modelling.** Our main focus of this thesis is to address this issue, as effective high-level contextual modelling generally leads to phenomenal performance boost. In detail, we firstly propose to inject global scene context into local labeling of patch-CNN. This global scene constraint is transferred from an non-parametric model and it is fused with local beliefs output by patch-CNN. In practice, such global context is very effective to mitigate the local ambiguity problem of local labeling achieved by patch-CNN, thus it significantly improves the parsing performance of patch-CNN. Experimentally, we also observe a significant **5%** and **8%** improvement over overall pixel accuracy on SiftFlow [1] and Barcelona dataset [2] respectively. Next, we propose and formulate two computational layers (DAG-RNN, CCN) to aggregate context for feature maps. These layers are placed on top of pre-trained CNN to distill context into feature maps so that their representative capability are significantly enhanced. Importantly, these layers are jointly learned with the rest segmentation network components. Our experiments demonstrate that these context aggregation layers are very essential to enhance the high-level parsing performance of segmentation networks. For examples, the introduction of DAG-RNN and CCN brings in a significant **3%** and **7%** IOU boost over competitive FCN baseline [21] on Pascal-Context [11] respectively. Meanwhile, we extensively compare our works against state-of-the-art context aggregation models quantitatively and qualitatively.
- **Imbalanced class distribution.** We propose two different approaches to address this challenge under different learning framework. In patch-based CNN training, we propose and discuss multiple “patch sampling” approach to generate training

batches, based on which patch-CNN focuses on learning robust representation and classifiers for semantic classes within specific occurrence frequency ranges. We then propose CNN-Ensemble that combines predictions of these complementary patch-CNN, and we demonstrate that CNN-Ensemble performs competitively well both for frequent and rare classes. Concretely in our experiments, we observe a significant **14%** and **8%** improvement in terms of average class accuracy (ACA) on SiftFlow [1] and Barcelona dataset [2] respectively. In the image-to-image learning paradigm (i.e. DAG-RNN and IFCN), we propose a novel class-weighted loss that distributes reasonably higher attention weight to infrequent classes based on their rareness magnitude. Thus, those infrequent class pixels are expected to be learned faster comparing to frequent classes. Our experiments show that the proposed class-weighted loss is a very economic as well as an effective way to boost the parsing accuracies of rare classes. For example, a significant **3.2%** ACA improvement is observed when the proposed loss is engaged in DAG-RNN training on challenging Pascal-Context dataset [11].

- **Lower-level information incorporation.** We address this issue by adapting pre-trained CNN using the proposed dense skip architecture. More specifically, we examine the limitations of current parametrization of skip layers, based on which we slightly modify its parametrization. In our dense skip architecture, rich set of feature maps ranging from low-level to high-level are incorporated into dense semantic prediction. This architecture modification greatly improves the capability of the corresponding segmentation network at detecting small-size objects as well as localizing region boundaries. Overall, we observe a promising **2.2%** and **1.9%** IOU boost for FCN [21] and DeepLab-v2 [22] when dense skip architecture is used.

6.2 Future works

Semantic segmentation (including scene segmentation and object segmentation) has witnessed significant progress after deep neural networks are incorporated into building segmentation models. It's quite obvious that the progress of segmentation community is expected to be highly correlated with the advancement of deep neural networks in the near future. On the other hand, the core of further improving the overall parsing accuracies of segmentation models is to effectively address the key challenges elaborated in this thesis. Beyond that, we may discuss the possible unfolding works in the foreseeable future.

- **Weakly-supervised or Unsupervised scene segmentation.** One important issue of fully-supervised scene segmentation is that pixel-wise human annotations are essential to train the segmentation models. Unfortunately, as pointed out by [11], it's extremely expensive to generate a dense as well as an accurate annotation map in practice. This also explains why the sizes of public scene segmentation datasets are relatively smaller comparing to those of popular object detection datasets. As we have very limited fully-annotated images and very large amount of weakly-annotated images (image tags, or object bounding boxes), it's interesting to explore how can we leverage weakly-annotated data in a self-learning manner to boost the parsing performance of current segmentation networks. It's even more interesting to explore how to build effective segmentation models that are less reliant on dense and accurate annotation maps.
- **Speeding up and compressing segmentation networks.** As current trend of building segmentation models is to primarily improving their parsing accuracies, researchers usually resort to use very deep pre-trained CNN (e.g. VGG-16 [13], ResNet [19]) to extract features or to develop extra networks to aggregate context (e.g. DAG-RNN, CCN, DeepLab [22]). These practices significantly degrade the

inference efficiency as well as increase the memory consumption of segmentation networks. Therefore, it is technically demanding to develop efficient as well as light segmentation models so that they can be used in resource-constrained devices in the future.

- **Instance-level semantic segmentation.** This is a combination of multiple interesting tasks that includes scene segmentation and object detection [72] [73] [150] [119]. Different from simply generating a semantic prediction map, instance-level semantic segmentation demands to differentiate each constituent object instances, if there is any. This task also shares great similarity with instance-level object segmentation [151] [152] [101]. Instance-level semantic segmentation has more promising applications as it provides even deeper scene understanding. For example, we are able to understand how many persons are present in the scene and how they are interacting with what objects, we may also be able to have a rough understanding towards each person's activity (pose) based on their instance segmentation masks. As stuff classes [153] are expected to provide extra contexts for object detection module, we may also expect that instance-level segmentation models, if properly designed, will improve object detection accuracies as well.

Author's publications

- **Bing Shuai**, Zhen Zuo, Bing Wang and Gang Wang, “Scene Segmentation with DAG-Recurrent Neural Network”, In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017. (To appear)
- **Bing Shuai***, Zhen Zuo*, Gang Wang and Bing Wang, “Dag-recurrent neural networks for scene labeling”, In *IEEE Proceedings on Computer Vision and Pattern Recognition (CVPR)*, pp. 3620-3629, 2016.
- **Bing Shuai**, Zhen Zuo, Gang Wang and Bing Wang, “Scene Parsing with Integration of Parametric and Non-parametric Models”, In *IEEE Transactions on Image Processing (TIP)*, vol. 25, no. 5, pp. 2379 - 2391, 2016.
- **Bing Shuai**, Gang Wang, Zhen Zuo, Bing Wang and Lifan Zhao, “Integrating parametric and non-parametric models for scene labeling”, In *IEEE Proceedings on Computer Vision and Pattern Recognition (CVPR)*, pp. 4249 - 4258, 2015.
- **Bing Shuai**, Zhen Zuo and Gang Wang, “Quaddirectional 2d-recurrent neural networks for image labeling”, In *IEEE Signal Processing Letters (SPL)*, vol. 22, no. 11, pp. 1990 - 1994, 2015.

* indicates equal contributions.

Author's other publications

- Abrar Abudl Nabi, **Bing Shuai** and Gang Wang, “Episodic CAMN: Contextual Attention-based Memory Networks for Scene Labeling”, In *IEEE Proceedings on Computer Vision and Pattern Recognition (CVPR)*, pp. 6278 - 6287, 2017.
- Ping Hu, **Bing Shuai** and Gang Wang, “Deep Level Sets for Salient Object Detection”, In *IEEE Proceedings on Computer Vision and Pattern Recognition (CVPR)*, pp. 540 - 549, 2017.
- Rahul Rama Varior, **Bing Shuai**, Jiwen Lu, Dong Xu, and Gang Wang, “A siamese long short-term memory architecture for human re-identification”, In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 135-153, 2016.
- Zhen Zuo, **Bing Shuai**, Gang Wang, Xiao Liu, Xingxing Wang and Bing Wang, “Learning Contextual Dependencies with Convolutional Hierarchical Recurrent Neural Networks”, vol. 25, no. 7, pp. 2983 - 2996, In *IEEE Transactions on Image Processing (TIP)*, 2016.
- Zhen Zuo, **Bing Shuai**, Gang Wang, Xiao Liu, Xingxing Wang, Bing Wang and Yushi Chen, “Convolutional recurrent neural networks: Learning spatial dependencies for image representation”, In *IEEE Proceedings on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 18 - 26, 2015.
- Zhen Zuo, Gang Wang, **Bing Shuai**, Lifan Zhao and Qingxiong Yang, “Exemplar based deep discriminative and shareable feature learning for scene image classification” In *Pattern Recognition (PR)*, vol. 48, no. 10, pp. 3004-3015, 2015.

- Zhen Zuo, Gang Wang, **Bing Shuai**, Lifan Zhao, Qingxiong Yang and Xudong Jiang, “Learning discriminative and shareable features for scene classification”, In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 552-568, 2014.

References

- [1] C. Liu, J. Yuen, and A. Torralba, “Nonparametric scene parsing: Label transfer via dense scene alignment,” in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 1972–1979, IEEE, 2009.
- [2] J. Tighe and S. Lazebnik, “Superparsing: scalable nonparametric image parsing with superpixels,” in *Computer Vision–ECCV 2010*, pp. 352–365, Springer, 2010.
- [3] J. Tighe and S. Lazebnik, “Finding things: Image parsing with regions and per-exemplar detectors,” in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 3001–3008, IEEE, 2013.
- [4] M. George, “Image parsing with a wide range of classes and scene-level context,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3622–3630, 2015.
- [5] S. L. Dockstader and A. M. Tekalp, “Multiple camera fusion for multi-object tracking,” in *Multi-Object Tracking, 2001. Proceedings. 2001 IEEE Workshop on*, pp. 95–102, IEEE, 2001.
- [6] B. Wang, G. Wang, K. Luk Chan, and L. Wang, “Tracklet association with online target-specific metric learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1234–1241, 2014.

- [7] M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, “A solution to the simultaneous localization and map building (slam) problem,” *IEEE Transactions on robotics and automation*, vol. 17, no. 3, pp. 229–241, 2001.
- [8] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. Montiel, “Towards semantic slam using a monocular camera,” in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 1277–1284, IEEE, 2011.
- [9] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in *International Conference on Machine Learning*, pp. 2048–2057, 2015.
- [10] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3156–3164, 2015.
- [11] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille, “The role of context for object detection and semantic segmentation in the wild,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 891–898, 2014.
- [12] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Semantic understanding of scenes through the ade20k dataset,” *arXiv preprint arXiv:1608.05442*, 2016.
- [13] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [14] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, “Imagenet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

- [15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, IEEE, 2009.
- [16] A. Graves, “Offline arabic handwriting recognition with multidimensional recurrent neural networks,” in *Guide to OCR for Arabic Scripts*, pp. 297–313, Springer, 2012.
- [17] A. Graves, A.-R. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 6645–6649, IEEE, 2013.
- [18] H. Caesar, J. Uijlings, and V. Ferrari, “Coco-stuff: Thing and stuff classes in context,” *arXiv preprint arXiv:1612.03716*, 2016.
- [19] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.
- [20] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.
- [21] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.
- [22] 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,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

REFERENCES

- [23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, pp. 1097–1105, 2012.
- [24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.
- [25] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” in *AAAI*, pp. 4278–4284, 2017.
- [26] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5987–5995, IEEE, 2017.
- [27] G. Huang, Z. Liu, and L. Weinberger, Kilian Q andfr van der Maaten, “Densely connected convolutional networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017.
- [28] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017.
- [29] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, “Semantic image segmentation via deep parsing network,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1377–1385, 2015.
- [30] G. Lin, C. Shen, A. van den Hengel, and I. Reid, “Efficient piecewise training of deep structured models for semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3194–3203, 2016.

- [31] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 41–48, 2016.
- [32] A. Sharma, O. Tuzel, and M.-Y. Liu, “Recursive context propagation network for semantic scene labeling,” in *Advances in Neural Information Processing Systems*, pp. 2447–2455, 2014.
- [33] W. Liu, A. Rabinovich, and A. C. Berg, “Parsenet: Looking wider to see better,” *International Conference on Learning Representation (ICLR) Workshop*, 2016.
- [34] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” *International Conference on Learning Representation (ICLR)*, 2016.
- [35] G. Lin, A. Milan, C. Shen, and I. Reid, “Refinenet: Multi-path refinement networks with identity mappings for high-resolution semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [36] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [37] V. Badrinarayanan, A. Kendall, and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation,” *arXiv preprint arXiv:1511.00561*, 2015.
- [38] H. Noh, S. Hong, and B. Han, “Learning deconvolution network for semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1520–1528, 2015.

- [39] V. Nekrasov, J. Ju, and J. Choi, “Global deconvolutional networks for semantic segmentation,” in *British Machine Vision Conference (BMVC)*, 2016.
- [40] p. Wang, p. Chen, y. Yuan, D. Liu, z. Huang, X. Hou, and G. Cottrell, “Understanding convolution for semantic segmentation,” *arXiv preprint arXiv:1702.08502*, 2017.
- [41] P. Krähenbühl and V. Koltun, “Efficient inference in fully connected crfs with gaussian edge potentials,” in *NIPS*, pp. 109–117, 2011.
- [42] 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.
- [43] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, “Conditional random fields as recurrent neural networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1529–1537, 2015.
- [44] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *Computer Vision–ECCV 2014*, pp. 740–755, Springer, 2014.
- [45] z. Wu, c. Shen, and A. v. d. Hengel, “Wider or deeper: Revisiting the resnet model for visual recognition,” *arXiv preprint arXiv:1611.10080*, 2016.
- [46] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6230 – 6239, 2017.
- [47] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, “Learning hierarchical features for scene labeling,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013.

REFERENCES

- [48] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich, “Feedforward semantic segmentation with zoom-out features,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3376–3385, 2015.
- [49] H. Caesar, J. Uijlings, and V. Ferrari, “Joint calibration for semantic segmentation,” in *BMVC*, 2015.
- [50] B. Shuai, Z. Zuo, G. Wang, and B. Wang, “Scene parsing with integration of parametric and non-parametric models,” *IEEE Transactions on Image Processing*, vol. 25, no. 5, pp. 2379–2391, 2016.
- [51] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [52] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 886–893, IEEE, 2005.
- [53] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” *Computer vision–ECCV 2006*, pp. 404–417, 2006.
- [54] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *Computer Vision (ICCV), 2011 IEEE international conference on*, pp. 2564–2571, IEEE, 2011.
- [55] E. Tola, V. Lepetit, and P. Fua, “A fast local descriptor for dense matching,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, IEEE, 2008.
- [56] P. Pinheiro and R. Collobert, “Recurrent convolutional neural networks for scene labeling,” in *Proceedings of The 31st International Conference on Machine Learning*, pp. 82–90, 2014.

- [57] L. Wang, T. Liu, G. Wang, K. L. Chan, and Q. Yang, “Video tracking using learned hierarchical features,” *Image Processing, IEEE Transactions on*, vol. 24, no. 4, pp. 1424–1435, 2015.
- [58] Z. Zuo, G. Wang, B. Shuai, L. Zhao, and Q. Yang, “Exemplar based deep discriminative and shareable feature learning for scene image classification,” *Pattern Recognition*, vol. 48, no. 10, pp. 3004–3015, 2015.
- [59] A. Criminisi and J. Shotton, *Decision forests for computer vision and medical image analysis*. Springer Science & Business Media, 2013.
- [60] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [61] J. Shotton, M. Johnson, and R. Cipolla, “Semantic texton forests for image categorization and segmentation,” in *Computer vision and pattern recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, IEEE, 2008.
- [62] P. Kotschieder, S. R. Bulò, H. Bischof, and M. Pelillo, “Structured class-labels in random forests for semantic image labelling,” in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2190–2197, IEEE, 2011.
- [63] S. R. Bulò and P. Kotschieder, “Neural decision forests for semantic image labelling,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 81–88, IEEE, 2014.
- [64] J. J. Lim, C. L. Zitnick, and P. Dollár, “Sketch tokens: A learned mid-level representation for contour and object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3158–3165, 2013.
- [65] P. Dollár and C. L. Zitnick, “Structured forests for fast edge detection,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1841–1848, 2013.

- [66] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, pp. 2169–2178, IEEE, 2006.
- [67] J. Yang, K. Yu, Y. Gong, and T. Huang, “Linear spatial pyramid matching using sparse coding for image classification,” in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 1794–1801, IEEE, 2009.
- [68] F. Perronnin, J. Sánchez, and T. Mensink, “Improving the fisher kernel for large-scale image classification,” *Computer Vision–ECCV 2010*, pp. 143–156, 2010.
- [69] Z. Zuo, G. Wang, B. Shuai, L. Zhao, Q. Yang, and X. Jiang, “Learning discriminative and shareable features for scene classification,” in *Computer Vision–ECCV 2014*, pp. 552–568, Springer, 2014.
- [70] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, “Learning deep features for scene recognition using places database,” in *Advances in neural information processing systems*, pp. 487–495, 2014.
- [71] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.
- [72] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448, 2015.
- [73] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in neural information processing systems*, pp. 91–99, 2015.

REFERENCES

- [74] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016.
- [75] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *European Conference on Computer Vision*, pp. 21–37, Springer, 2016.
- [76] B. Shuai, Z. Zuo, B. Wang, and G. Wang, “Dag-recurrent neural networks for scene labeling,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3620–3629, 2016.
- [77] E. Shelhamer, J. Long, and T. Darrell, “Fully convolutional networks for semantic segmentation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 4, pp. 640–651, 2017.
- [78] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, “Joint training of a convolutional network and a graphical model for human pose estimation,” in *Advances in neural information processing systems*, pp. 1799–1807, 2014.
- [79] A. Toshev and C. Szegedy, “Deeppose: Human pose estimation via deep neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1653–1660, 2014.
- [80] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. V. Gehler, and B. Schiele, “Deepcut: Joint subset partition and labeling for multi person pose estimation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4929–4937, 2016.
- [81] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele, “Deep-ercut: A deeper, stronger, and faster multi-person pose estimation model,” in *European Conference on Computer Vision*, pp. 34–50, Springer, 2016.

- [82] M. Mostajabi and I. Gholampour, “A robust multilevel segment description for multi-class object recognition,” *Machine Vision and Applications*, vol. 26, no. 1, pp. 15–30, 2015.
- [83] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” *arXiv preprint arXiv:1706.05587*, 2017.
- [84] F. Yu, V. Koltun, and T. Funkhouser, “Dilated residual networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [85] J. Shotton, J. Winn, C. Rother, and A. Criminisi, “Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation,” in *Computer Vision–ECCV 2006*, pp. 1–15, Springer, 2006.
- [86] A. Roy and S. Todorovic, “Scene labeling using beam search under mutex constraints,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1178–1185, 2014.
- [87] Y. Zhang and T. Chen, “Efficient inference for fully-connected crfs with stationarity,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 582–589, IEEE, 2012.
- [88] P. Kohli, P. H. Torr, *et al.*, “Robust higher order potentials for enforcing label consistency,” *International Journal of Computer Vision*, vol. 82, no. 3, pp. 302–324, 2009.
- [89] P. Márquez-Neila, P. Kohli, C. Rother, and L. Baumela, “Non-parametric higher-order random fields for image segmentation,” in *Computer Vision–ECCV 2014*, pp. 269–284, Springer, 2014.
- [90] X. He, R. S. Zemel, and M. Carreira-Perpinan, “Multiscale conditional random fields for image labeling,” in *Computer vision and pattern recognition, 2004. CVPR*

REFERENCES

2004. *Proceedings of the 2004 IEEE computer society conference on*, vol. 2, pp. II–695, IEEE, 2004.
- [91] D. Eigen and R. Fergus, “Nonparametric image parsing using adaptive neighbor sets,” in *Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on*, pp. 2799–2806, IEEE, 2012.
- [92] G. Singh and J. Kosecka, “Nonparametric scene parsing with adaptive feature relevance and semantic context,” in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 3151–3157, IEEE, 2013.
- [93] F. Tung and J. J. Little, “Collageparsing: Nonparametric scene parsing by adaptive overlapping windows,” in *Computer Vision–ECCV 2014*, pp. 511–525, Springer, 2014.
- [94] J. Yang, B. Price, S. Cohen, and M.-H. Yang, “Context driven scene parsing with attention to rare classes,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 3294–3301, IEEE, 2014.
- [95] S. Gould and Y. Zhang, “Patchmatchgraph: Building a graph of dense patch correspondences for label transfer,” in *Computer Vision–ECCV 2012*, pp. 439–452, Springer, 2012.
- [96] S. Gould, J. Zhao, X. He, and Y. Zhang, “Superpixel graph label transfer with learned distance metric,” in *Computer Vision–ECCV 2014*, pp. 632–647, Springer, 2014.
- [97] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. CRC Press, 2012.
- [98] J. Shotton, T. Sharp, P. Kohli, S. Nowozin, J. Winn, and A. Criminisi, “Decision jungles: Compact and rich models for classification,” in *Advances in Neural Information Processing Systems*, pp. 234–242, 2013.

REFERENCES

- [99] L. Breiman, “Bagging predictors,” *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.
- [100] Y. Freund, R. Schapire, and N. Abe, “A short introduction to boosting,” *Journal-Japanese Society For Artificial Intelligence*, vol. 14, no. 771-780, p. 1612, 1999.
- [101] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.
- [102] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, “Dssd: Deconvolutional single shot detector,” *arXiv preprint arXiv:1701.06659*, 2017.
- [103] J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” *arXiv preprint arXiv:1612.08242*, 2016.
- [104] Q. Li, J. Wang, Z. Tu, and D. P. Wipf, “Fixed-point model for structured labeling,” in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 214–221, 2013.
- [105] Z. Tu, “Auto-context and its application to high-level vision tasks,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, IEEE, 2008.
- [106] O. Irsoy and C. Cardie, “Opinion mining with deep recurrent neural networks,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 720–728, 2014.
- [107] M. Sundermeyer, T. Alkhouli, J. Wuebker, and H. Ney, “Translation modeling with bidirectional recurrent neural networks,” in *EMNLP*, pp. 14–25, 2014.
- [108] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in neural information processing systems*, pp. 3104–3112, 2014.

- [109] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [110] F. Visin, K. Kastner, K. Cho, M. Matteucci, A. Courville, and Y. Bengio, “Renet: A recurrent neural network based alternative to convolutional networks,” *arXiv preprint arXiv:1505.00393*, 2015.
- [111] Z. Zuo, B. Shuai, G. Wang, X. Liu, X. Wang, B. Wang, and Y. Chen, “Convolutional recurrent neural networks: Learning spatial dependencies for image representation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 18–26, 2015.
- [112] B. Shuai, Z. Zuo, and G. Wang, “Quaddirectional 2d-recurrent neural networks for image labeling,” *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 1990–1994, 2015.
- [113] K. S. Tai, R. Socher, and C. D. Manning, “Improved semantic representations from tree-structured long short-term memory networks,” *arXiv preprint arXiv:1503.00075*, 2015.
- [114] X. Zhu, P. Sobihani, and H. Guo, “Long short-term memory over recursive structures,” in *International Conference on Machine Learning*, pp. 1604–1612, 2015.
- [115] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu, “Advances in optimizing recurrent networks,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 8624–8628, IEEE, 2013.
- [116] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 41–48, 2016.

REFERENCES

- [117] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan, “Pixelnet: Towards a general pixel-level architecture,” *arXiv preprint arXiv:1609.06694*, 2016.
- [118] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, “Hypercolumns for object segmentation and fine-grained localization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 447–456, 2015.
- [119] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.
- [120] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [121] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1717–1724, 2014.
- [122] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1701–1708, 2014.
- [123] N. Zhang, M. Paluri, M. Ranzato, T. Darrell, and L. Bourdev, “Panda: Pose aligned networks for deep attribute modeling,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1637–1644, 2014.
- [124] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Computer Vision—ECCV 2014*, pp. 818–833, Springer, 2014.
- [125] J. Hu, J. Lu, and Y.-P. Tan, “Discriminative deep metric learning for face verification in the wild,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 1875–1882, IEEE, 2014.

REFERENCES

- [126] A. Vedaldi and K. Lenc, “Matconvnet: Convolutional neural networks for matlab,” in *Proceedings of the 23rd ACM international conference on Multimedia*, pp. 689–692, ACM, 2015.
- [127] A. Coates and A. Y. Ng, “Learning feature representations with k-means,” in *Neural networks: Tricks of the trade*, pp. 561–580, Springer, 2012.
- [128] A. Oliva and A. Torralba, “Building the gist of a scene: The role of global image features in recognition,” *Progress in brain research*, vol. 155, pp. 23–36, 2006.
- [129] F. Tung and J. J. Little, “Scene parsing by nonparametric label transfer of content-adaptive windows,” *Computer Vision and Image Understanding*, vol. 143, pp. 191–200, 2016.
- [130] C. Gatta, A. Romero, and J. van de Veijer, “Unrolling loopy top-down semantic feedback in convolutional deep networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 498–505, 2014.
- [131] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier, “Large-scale image retrieval with compressed fisher vectors,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 3384–3391, IEEE, 2010.
- [132] B. Shuai, G. Wang, Z. Zuo, B. Wang, and L. Zhao, “Integrating parametric and non-parametric models for scene labeling,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4249–4258, 2015.
- [133] J. L. Elman, “Finding structure in time,” *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.
- [134] P. J. Werbos, “Backpropagation through time: what it does and how to do it,” *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

- [135] J. S. Yedidia, W. T. Freeman, and Y. Weiss, “Understanding belief propagation and its generalizations,” *Exploring artificial intelligence in the new millennium*, vol. 8, pp. 236–239, 2003.
- [136] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky, “A new class of upper bounds on the log partition function,” *Information Theory, IEEE Transactions on*, vol. 51, no. 7, pp. 2313–2335, 2005.
- [137] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” in *Advances in neural information processing systems*, pp. 3320–3328, 2014.
- [138] Q. V. Le, N. Jaitly, and G. E. Hinton, “A simple way to initialize recurrent networks of rectified linear units,” *arXiv preprint arXiv:1504.00941*, 2015.
- [139] J. Dai, K. He, and J. Sun, “Convolutional feature masking for joint object and stuff segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3992–4000, 2015.
- [140] S. Xie, X. Huang, and Z. Tu, “Convolutional pseudo-prior for structured labeling,” *arXiv preprint arXiv:1511.07409*, 2015.
- [141] W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki, “Scene labeling with lstm recurrent neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3547–3555, 2015.
- [142] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in *International Conference on Machine Learning*, pp. 1310–1318, 2013.
- [143] S. Song, S. P. Lichtenberg, and J. Xiao, “Sun rgb-d: A rgb-d scene understanding benchmark suite,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 567–576, 2015.

REFERENCES

- [144] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *ICML*, pp. 448–456, 2015.
- [145] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from rgb-d images,” in *European Conference on Computer Vision*, pp. 746–760, Springer, 2012.
- [146] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell, “A category-level 3d object dataset: Putting the kinect to work,” in *Consumer Depth Cameras for Computer Vision*, pp. 141–165, Springer, 2013.
- [147] J. Xiao, A. Owens, and A. Torralba, “Sun3d: A database of big spaces reconstructed using sfm and object labels,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1625–1632, 2013.
- [148] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, “Simultaneous detection and segmentation,” in *European Conference on Computer Vision*, pp. 297–312, Springer, 2014.
- [149] G. Ghiasi and C. C. Fowlkes, “Laplacian pyramid reconstruction and refinement for semantic segmentation,” in *European Conference on Computer Vision*, pp. 519–534, Springer, 2016.
- [150] J. Dai, Y. Li, K. He, and J. Sun, “R-fcn: Object detection via region-based fully convolutional networks,” in *Advances in neural information processing systems*, pp. 379–387, 2016.
- [151] J. Dai, K. He, and J. Sun, “Instance-aware semantic segmentation via multi-task network cascades,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3150–3158, 2016.

REFERENCES

- [152] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, “Fully convolutional instance-aware semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4438–4446, 2017.
- [153] S. Brahmabhatt, H. I. Christensen, and J. Hays, “Stuffnet: Using ?stuff?to improve object detection,” in *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pp. 934–943, IEEE, 2017.