Improving Semantic Image Segmentation by Object Localization

by

Zichen Zhang

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Semantic segmentation is about classifying every pixel in an image. In recent years, methods based on Fully Convolutional Networks (FCN) have dominated this field in terms of segmentation accuracy. We are interested in tackling the challenges that these methods are faced with. First, it is expensive to acquire pixel level labels to train the network. Second, FCN often has trouble with data that present imbalanced positive and negative samples. This issue often comes up in domains such as medical imaging and satellite imagery analysis, where the object of interest can be very small. The large number of negative samples can overwhelm the positive samples during training, leading to a biased representation learned by the network. In this thesis, we investigate how an object localization mechanism can address these two challenges.

We propose an end-to-end neural network that improves the segmentation accuracy of FCN by incorporating an object localization unit. This network performs object localization first, which is then used as a cue to guide the training of the segmentation network. The two steps share convolutional features. This allows us to leverage object detection labels to help with the training of the segmentation network, alleviating the need for large-scale pixel level labels. To avoid applying max pooling on object proposals that limits the spatial accuracy, we introduce a new type of convolutional layer named ROI convolution. It applies convolution directly on the object proposals in one shot, without the need of passing them individually through the downstream network. We show that this layer is differentiable therefore allowing the network to be trained end-to-end.

To demonstrate the efficacy of our method, we apply it to the problem of medical image segmentation. With the object localization unit, our method performs well despite the high class imbalance and it outperforms existing methods on small object segmentation. To understand further about the proposed method and the impact of ROI convolution, we also conducted ablation studies and experimented on an endoscopic image dataset with balanced data.

Acknowledgements

First of all, I would like to thank my supervisors Dr. Dana Cobzas and Dr. Martin Jagersand for their guidance and support. In particular, I want to thank Dana for introducing me to the field of medical image analysis and providing hands-on guidance on research throughout my study. I want to thank Martin for giving me the freedom to wander around different projects and for his patience and knowledgeable advice along the way. I am very grateful for Dr. Nilanjan Ray for serving on my supervisory committee and for his time reviewing my thesis.

I would also like to thank the members at the vision and robotics labs: Masood, Camilo, Xuebin, Shida, Min, Menna, Rong, Jun, Oscar, Mona, Ankush, Xi. They have made my life in and out of the lab so much fun. I am looking forward to continuing this fun journey with those of them who would remain in the lab.

A special thanks goes to NSERC, Alberta Inovates and FGSR for offering me the scholarships.

I want to take this opportunity to thank my Mom, Ping Ding and my Dad, Baoping Zhang, for their endless love and support, and for allowing me to pursue my study abroad. I also want to thank my grandmother who passed away last year, for taking care of me since my childhood. I am blessed to have been so close to her and yet I am deeply sorry for not being at her bedside in her last days. Last but not least, I would like to thank my wife Qin Tao for her love, support, loving personality and patience. I am beyond fortunate to have met her.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Deep learning has revolutionized the field of computer vision in the past few years. In particular, Convolutional Neural Networks (CNNs) has become a standard architecture and achieved great success in many computer vision tasks, including image classification, object detection and semantic segmentation.

Image classification is about predicting the classes for the objects contained in the image. Object detection adds localization on top of image classification. The goal is to not only predict the class for each object, but also find its bounding box. This thesis is focused on the task of semantic segmentation, which is about classifying every pixel in an image. It can be viewed as a pixel-level classification task, whereas image classification and object detection perform image-level and region-level classification, respectively. Despite the success of deep learning in semantic segmentation, there remains some open challenges that CNNs often have issues with, including:

1. The lack of labels for training: the advancement of deep learning in the past decade is partially due to the large-scale public datasets such as ImageNet [48], MS COCO [36]. These datasets provide millions of labeled images that allow training very deep neural networks without over-fitting. Acquiring the annotations for segmentation is a tedious and time consuming process. The annotations for large-scale datasets are typically obtained through crowd-sourcing. ImageNet and MS COCO both used Amazon Mechanical Turk (AMT) [8] to pay on-line workers

to annotate the images. However, crowd-sourcing is not applicable in some domains. For example, annotating medical imaging data often requires domain-specific expertise. Although there have been attempts of crowd-sourcing medical image annotation [17], [40], they were limited to particular use cases. The study in [17] was only performed on image classification. And Maier-Hein *et al.* [40] only studied the segmentation of surgical instruments on Endoscopic data. Another concern is the highly private nature of medical records, which limits the type of data that can be posted on-line [42]. In general, the labeling work needs a doctor or other well-trained professionals to perform for long hours. There are often inter-expert variations in the labels, which would take extra time to reconcile.

2. Imbalanced Data: CNNs typically need a reasonable number of positive and negative training samples to learn the representation. They work best when the ratio of positive and negative samples are balanced. However, this is not the case in some applications such as medical imaging and satellite imagery analysis, where the object of interest can be very small. The negative samples will overwhelm the positive samples during training, leading to a biased representation learned by the network.

In this thesis, we propose an algorithm that aims at tackling these challenges. The main contribution of this thesis is that we propose a novel convolutional neural network architecture that incorporates an object localization unit to improve semantic segmentation accuracy. We introduce a new type of convolutional layer named ROI convolution. This layer is simple, easy to use, yet effective. We show that this layer is differentiable therefore allowing the network to be trained end-to-end. To demonstrate the efficacy of our method, we apply it to the problem of medical image segmentation. We test our method on an ultrasound image dataset where the object of interest is small. We show that with the localization unit, our method performs well despite the high class imbalance and it outperforms the conventional methods. To understand further about the impact of ROI convolution, We also conducted ablation studies and experimented on an endoscopic image dataset with objects of regular size.

The rest of this thesis is organized as follows. Chapter 2 gives an overview of deep learning in semantic segmentation and object detection and covers some methods in medical image segmentation. We summarize the related work, introduce the key concepts that lead to our work and explain where the proposed method fits in the literature. Chapter 3 describes our proposed network and introduces the new layer ROI convolution. Chapter 4 shows the experiments of applying our method on two image segmentation tasks and the ablation study on the impact of ROI convolution and the localization mechanism. Chapter 5 concludes the thesis and points out the future directions of this line of work.

Chapter 2

Background and Related Works

In this chapter, we present an overview of deep learning algorithms in semantic segmentation and object localization. The methods in medical image segmentation are briefly summarized as well. We cover the related works and introduce the key concepts of the deep learning methods that are most relevant to our proposed work. In the end, we further explains the connection of our proposed method to the literature and highlight the major differences.

2.1 General Architectures of CNN

Convolutional Neural Network (CNN) is a class of neural network architectures that makes heavy use of convolutional layers. It is effective in learning the hidden features of data, especially in computer vision tasks. Its popularity today is largely due to the success of AlexNet [31]. AlexNet was a CNN that made a major breakthrough in 2012. Prior to that year, the traditional algorithms that relied on hand-engineered features had long been the state-of-the-art. AlexNet won the ImageNet 2012 competition by a large margin, outperforming all the traditional algorithms. Since then, the computer vision community has witnessed a dramatic paradigm shift from algorithms that use hand-engineered features to CNN-based algorithms. Many CNN architectures have been proposed during the process. Some of the most popular ones include VGG16 [49], GoogleNet [50], ResNet [23]. We show the architecture of VGG16 in Fig. 2.1, as it is the most relevant to our proposed algorithm.

These CNNs were originally developed for image classification but then

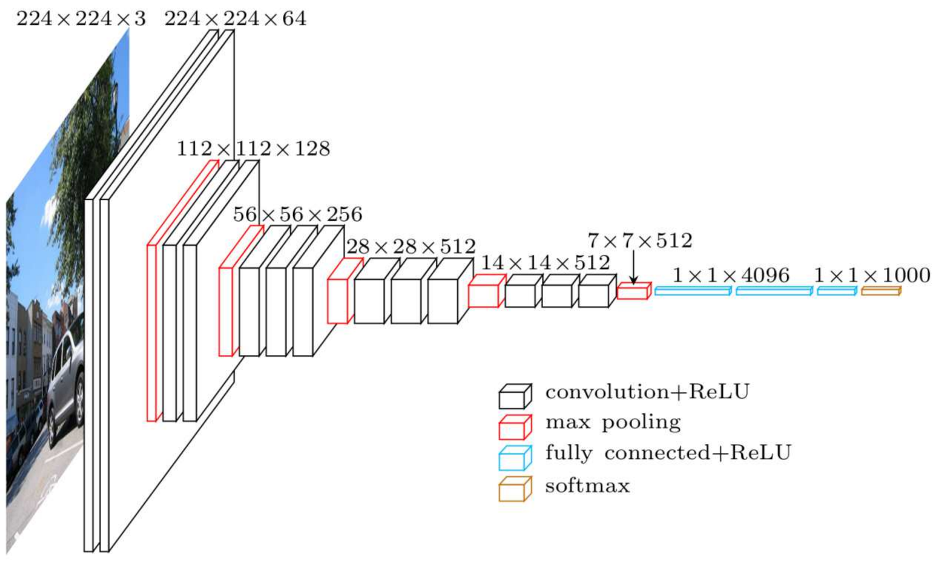


Image credit: https://recodbr.files.wordpress.com/2016/07/talk2cordunicamp2016.pdf

Figure 2.1: VGG16 architecture

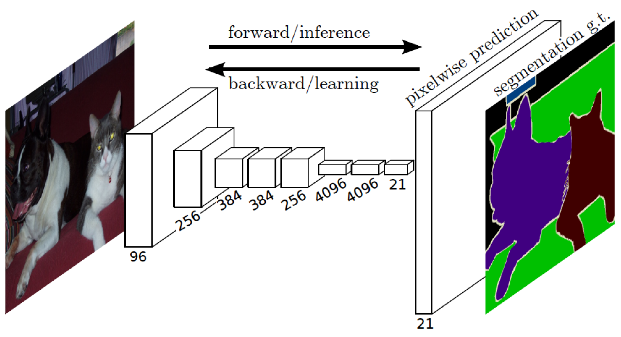
adapted and applied to many other applications as the backbone network. When transferring the architecture to other tasks, the top layers of the network are often adapted as they carry high-level semantic information that is specific to the task. The bottom layers are kept intact on the other hand, since the low level features are found to be very similar across different tasks [53]. It has become a standard to start from a network that has been pre-trained on ImageNet data and fine-tune the network to the data for the specific application.

2.2 Semantic Segmentation

2.2.1 Fully Convolutional Networks (FCN)

Fully Convolutional Network (FCN) [39] is the de-facto network architecture in semantic segmentation and has been the basis for many advanced network models since its release. The architecture is shown in Fig. 2.2.

The architecture of FCN is adapted from the CNNs as discussed in Section 2.1. The top layers of CNNs for image classification are typically fully connected (FC) layers that transform the features to a fixed size vector while



(a) FCN

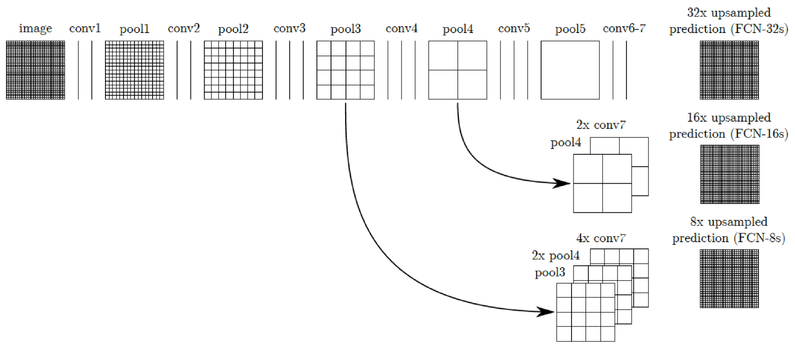


image
conv1
pool1
conv2
pool2
conv3
pool3
conv4
pool4
conv5
pool5
conv6-7
32x upsampled prediction (FCN-32s)
2x conv7
pool4
16x upsampled prediction (FCN-16s)
4x conv7
2x pool4
pool3
8x upsampled prediction (FCN-8s)

(b) Skip Architecture

Figure 2.2: Architecture of FCN [39]

capturing global context. The problem is that these FC layers expect the input to be of a certain size. As a result, the input images for these CNNs have to be resized to conform to the same size. For image classification, the resizing is not an issue since we only want to predict a class at the image-level. However, for image segmentation where we need pixel-level accuracy, the resizing would not be ideal. To enable the network to take images of various size and make pixel-wise classification, FCN replaces the fully connected layers with convolutional layers. The name of FCN reflects this change.

The main idea is to first pass the input image through a downsampling path to get an encoded representation. This representation then gets sent through an upsampling path to get the score map of the same size as the input. The score map is a probability map with several channels, each of which corresponds to one class. It represents the probability of each pixel belonging to a particular class. The segmentation result will be the class that gets the highest probability for each pixel. To improve the spatial accuracy, skip layers shown in Fig.2.2b were used where the feature maps at different scales were put together by element-wise sum. The downsampling path contains convolutional layers and pooling layers, whereas the upsampling path contains transposed convolutional layers that convert the condensed feature maps back to the size of the input image.

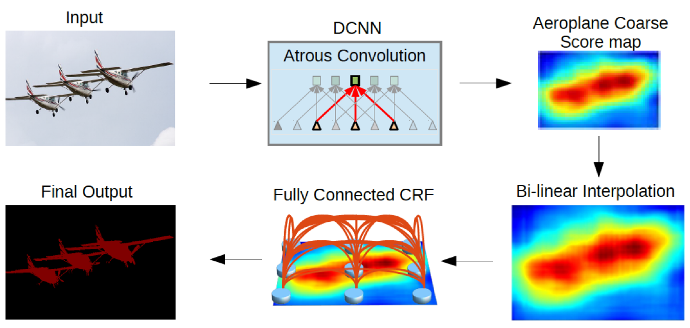
2.2.2 Advanced Variants of FCN

Since the inception of FCN, there have been a lot of work that builds on top of it, some of which are shown in Fig. 2.3.

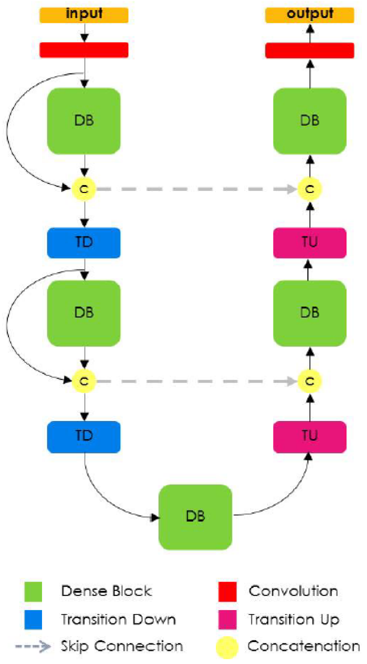
DeepLab [9] added atrous convolution to a convolutional network backbone like VGG16 [49] or ResNet [23], to handle objects at multiple scales. It applied a fully connected Conditional Random Field (CRF) as a post-processing step to refine the output. The workflow is shown in Figure 2.3a. Dilation Network [54] designed a “multi-scale context aggregation" module that makes use of multiple dilated convolutional layers to capture more contextual information from the image. The “dilated convolution" is the same as the “atrous convolution", but the author argued that “dilated convolution" is a better term to use. We refer the readers to the original paper for details.

U-Net [47], as shown in Figure 2.3c, was developed specifically for medical image segmentation. The network has a U shape architecture where it tried to make upsampling and downsampling streams symmetrical, with skip connections that concatenates the feature maps from downsampling layers to those in the upsampling layers.

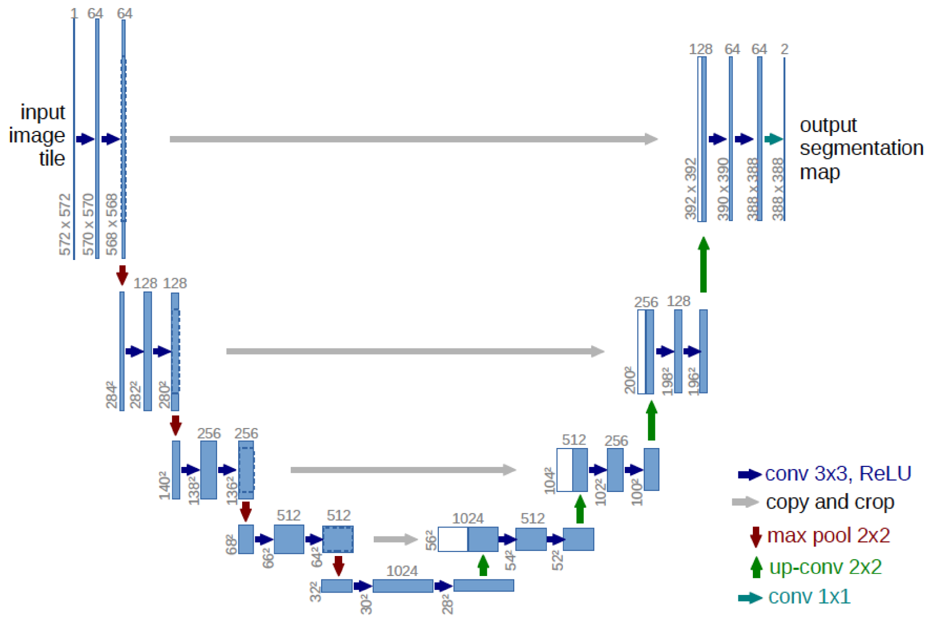
Simon et al. [26] adapted DenseNet [24] for semantic segmentation, where they employed dense skip connections, shown in Fig. 2.3b. Their network was also in a U shape. The dense skip connections were added to the downsampling stream only. Similar to U-Net, there were also skip connections connecting downsampling and upsampling streams. Unlike the previous works where the features maps were added element-wise, they concatenated the feature maps.



(a) DeepLab [9]



(b) Tiramisu Network [26]



(c) U-Net [47]

Figure 2.3: Varaints of FCN

2.2.3 Medical Image Segmentation

The majority of the work described in the last section were on natural images. Although they were designed to be general-purpose, their architectures often have to be adapted when applied to medical images, to account for the different characteristics of the data. Unlike natural images which are two dimensional (2D) colored images, medical images often come as three dimensional (3D) grayscale images and may contain multiple modalities. They take more time to annotate compared to natural images thus the number of available training images are fewer. Another key difference is that many tasks in medical imaging involve segmenting small regions. This requires deliberate designs of the neural network to avoid loosing the features of these regions as the feature maps get condensed through the layers. We summarize some existing works in medical image segmentation in this section.

Traditional methods: A nice summary of the traditional methods prior to deep learning was given in [21]. One class of the methods is based on discriminative models. Hand-designed features incorporating domain priors are generated for each voxel. A classifier then takes these features as input to classify each voxel. The popular choices of classifiers include SVM [4] and random forest [18]. The other class is based on variational methods, including Chan and Vesse [6], level-set [11], graphcut [52], *etc.*

Deep learning methods: A survey on recent deep learning methods on medical image analysis has been conducted in [37]. Segmentation was found to be the most common subject out of all the analysis tasks.

In terms of the size of the input data, deep learning segmentation methods can be divided into two classes: patch-based and image-based methods. Patch-based methods typically train a CNN on small patches (either 2D or 3D) sampled from the full-size images [21], [28], [43]. The output for each patch is the predicted class for the central pixel/voxel. The benefit of this approach is that it can control the number of positive and negative data in the input patches via different sampling methods. This allows them to avoid the issues of imbalanced data during network training. But a patch does not contain the global context. This is generally compensated for by using patches of multiple sizes in multiple streams and merge the features of the streams. On the other hand, image based methods directly apply an FCN on the original full size images. Brosch *et al*. developed a shallow encoder-decoder network [5], which consisted of two convolutional layers and two transposed convolutioanal layers. There were also deeper networks including U-Net described in the last section and its 3D extension [10].

There are various ways of incorporating the information of the third dimension and different modalities. **2D methods**: U-Net worked on 2D grayscale images. Brosch *et al*. used three-channel input data where the channels are from three modalities of the same 2D slice. **2.5D methods**: one can stack up three 2D slices along the Z-axis as in [43], or slices on the three axes as in the tri-planar approaches [32], [43]. **3D methods**: methods that directly applies on 3D voxels [10], [28]. In our work, we focus on the 2D methods.

2.2.4 Limitations of Existing Methods

Most of the image-based neural networks are trained with the same cross-entropy loss function that essentially makes the network learn a classifier for each pixel and average or sum up the loss for all pixels. This makes it hard to train the network when the object to segment is small. One can use the patch-base methods, where patches sampled from the original images were used as the input to the network [21], [29]. This alleviated the data imbalance problem. But on the downside, it caused computational overhead at inference time and introduced more hyper-parameters such as the size of the patch and the method of sampling patches for training. Another issue is that a small patch may lead to a loss of the context around the object which may be necessary for segmentation. Although this can be compensated for by sampling patches of various size and merging their feature maps at some point in the network [21], it again requires a lot of engineering effort to tune for the best setup. Our work aims to avoid this hand-engineering step with an algorithm inspired by object detection, described in Chapter 3.

2.3 Object Detection

Intuitively, humans perform semantic segmentation by first attending to some rough region where each pixel inside that region is then classified. In other words, object localization/detection can be a first step for accurate semantic segmentation. In this section, we cover the basics and related works in object detection literature.

2.3.1 Region Proposal Network (RPN)

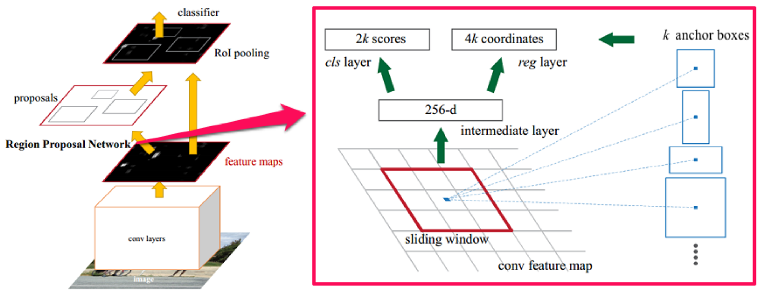
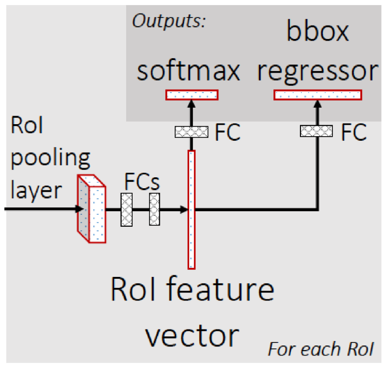


Image Credit: `https://andrewliao11.github.io/images/faster_rcnn/faster_rcnn_netwrok.png`

(a) Region Proposal Network



(b) Fast R-CNN

Figure 2.4: The architecture of Faster R-CNN [45]

The most widely adopted deep learning method for object detection is Faster R-CNN [45]. Its main architecture is named Region Proposal Network (RPN), shown in Fig. 2.4a. RPN takes an image and predicts a number of bounding boxes of the objects of interest. The workflow of RPN is as follows: 1. the image first gets passed through a backbone network to extract feature maps; 2. a small fully convolutional network is applied on the feature maps from step 1, trained with regression loss on the bounding box parameters and classification loss on the objectness of the object. Its output is the feature maps for these bounding boxes called region of interests (ROIs), and the objectness scores(background or foreground) associated with each ROI. 3. a pooling layer called ROI-Pooling is applied to each ROI to transform them into fix-size feature maps as the inputs for the downstream network.

The downstream network is the fast R-CNN network [19], used for detection, as shown in Fig. 2.4b. It is a small fully connected network that perform region-wise classification and regression. The feature maps from every ROI are first passed through two shared fully connected layers. Then the classification branch uses a fully connected layer to predict the class of the ROI, and the regression branch uses a fully connected layer to refine the location of the ROI.

2.3.2 ROI-Pooling and ROI-Align

The key step connecting the two sub-networks in Faster R-CNN is the ROI-pooling layer [19]. ROI-pooling layer applies max pooling to each ROI to convert them into fixed-size inputs. For a feature map of size h x w, and a pooling layer with the desired output size H x W, the feature map will be divided into a grid with bin size h/H x w/W. Max pooling is applied in each bin. The problem with ROI pooling is that it involves quantization in two steps: First, it computes the ground truth bounding box coordinate on the feature map as [x/stride], x is the original coordinate in the input image space, stride is the ratio between the size of the original input image and the feature map, [.] denotes rounding which rounds the float-point coordinate to the closest integer. Second, dividing the feature map into the bins also involves quantization.

These quantizations cause an alignment inaccuracy which inevitably hurts the localization performance. ROI-Align [22] was introduced to reduce this alignment error. The idea is to remove all the quantizations in ROI-pooling. It keeps the floating-point coordinate and uses regular sample location in each bin (either taking the value from the center pixel, or bilinear sampling at four regularly sampled locations) to compute the feature instead of using all the pixels inside the bin. It has significantly improved the overall detection performance.

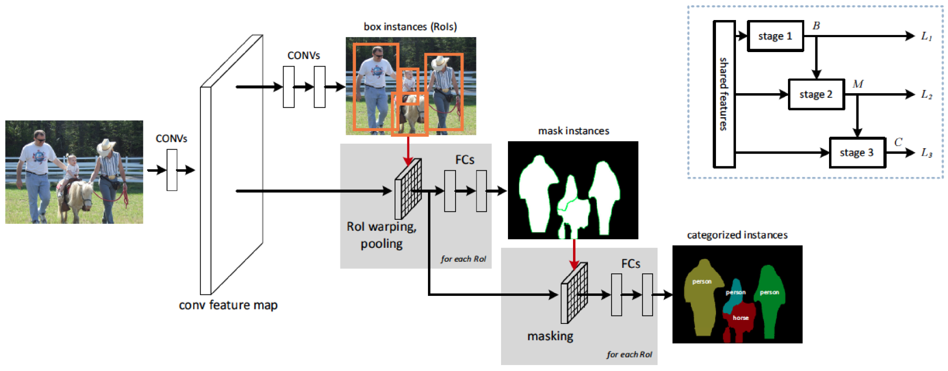
However, there are two general problems with pooling: 1. It loses spatial information. Pooling can have different size and strides. The pools overlap when the stride is smaller than the pool size. However, most CNNs use non-overlapping pools, where some spatial information is thrown away. The formulation makes the resulting features invariant to translations, which is necessary for the classification task, but it leads to a negative impact on tasks requiring pixel-level accuracy like detection and segmentation. This was an issue discussed by many. Geoff Hinton mentioned it recently in his online Q&A session [1], "The pooling operation used in convolutional neural networks is a big mistake and the fact that it works so well is a disaster. If the pools do not overlap, pooling loses valuable information about where things are. We need this information to detect precise relationships between the parts of an object." 2. Bin collapsing problem: ROI pooling is ill-defined if the desired output size is bigger than the input. It causes collapsing bins [55] where the output features are not discriminative.

2.3.3 ROI Convolution

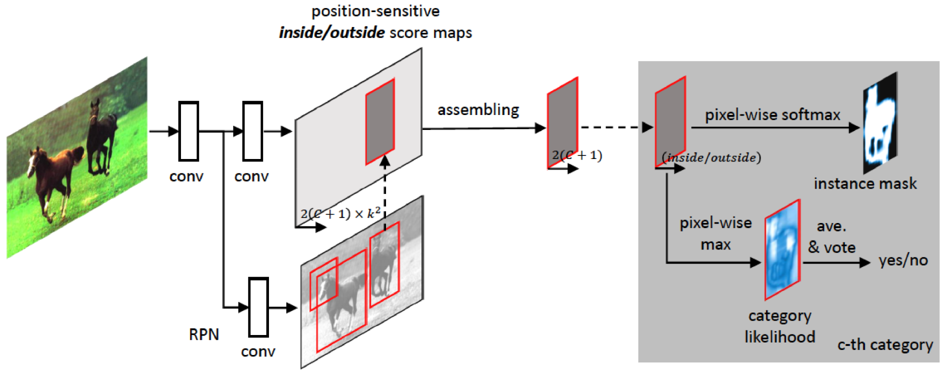
Dong et al. [7] proposed ROI convolution to speed up the network at inference time for face detection. They first used a conventional boosting cascade detector to generate ROIs. Then a cascaded CNN for face detection is applied only on the input inside the ROIs. The ROI convolution was excluded from the training of the network. During training time, neither boosting cascade detector or ROI convolution were used. There seemed to be duplicated computations in that the cascaded CNN already included a RPN which produced ROIs. But these ROIs never got used in the ROI convolution. During test time, all the convolutional layers are replaced with ROI convolutional layers, including those in the RPN, using the ROIs generated by the boosting detector. ROI convolution was viewed only as a means to improve computational efficiency. We will show in Chapter 4 that using ROI convolution in training time can bring significant benefits.

2.4 Instance Segmentation: The Combination of Segmentation and Detection

Instance segmentation deals with the problem of object-level pixel-wise classification. It combines object detection and segmentation. It is similar to object detection in that each object needs to be labeled separately. But for each object instance, instead of predicting the bounding box, it requires predicting the segmentation mask. The workflow can be thought of as object localization followed by semantic segmentation on each potential object instance. Because of its similarity to object detection, many works in instance segmentation re-purposed Faster R-CNN architecture by replacing the ROI classification module with pixel-wise classification module.



(a) MNC



(b) FCIS

Figure 2.5: The architecture of MNC [14] and FCIS [33]

Multi-task Network Cascades (MNC) [14] and Fully Convolutional Instance-aware Semantic Segmentation (FCIS) [33] are the winning methods of COCO

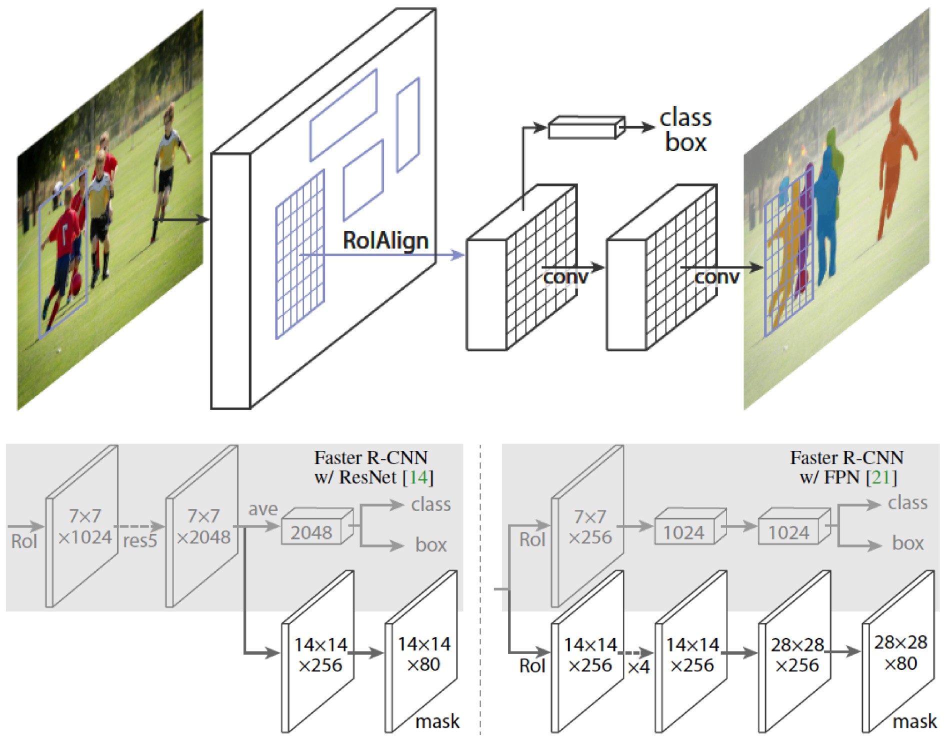


Figure 2.6: The architecture of Mask R-CNN [22]

segmentation challenge 2015 and 2016, respectively. MNC as shown in Fig. 2.5a is a cascaded network architecture that involves three steps. The first step uses RPN to predict the ROIs. The second step predicts semantic mask within each ROI. The third step classifies these masks. Step two and three are both performed on each ROI. Instead of dividing the task into multiple stages, FCIS performs mask prediction and classification simultaneously, as shown in Fig. 2.5b. Instead of extracting the features of each ROI with pooling, it predicts the inside and outside probability maps for each cell in a pre-defined grid of the ROIs and assembles the features for each cell.

Mask RCNN [22] is the state-of-the-art method in instance segmentation. Its architecture is shown in Figure 2.6. It first predicts the bounding box of each object instance in the image, then these ROIs are fed into two parallel branches. One of the branches predicts the class of each ROI and refines its location, same as in Faster R-CNN; the other branch is called mask branch, which predicts pixel-wise labels inside each ROI. The network was trained end-to-end with the multi-task loss of regression and classification on the bounding box, and a segmentation loss which is a binary cross-entropy loss on the ground-truth class versus the background class. It was shown that this multi-task loss improved the accuracy compared to training for each task in cascade. ROI-align layer was introduced in this work for accurate localization. Compared to ROI-pooling, or a similar layer called ROI-warping proposed in MNC, ROI-align significantly improves the segmentation accuracy. It indicates that precise bounding box localization is a key to accurate downstream segmentation. However, as discussed in the previous section, the use of pooling reduced the resolution of the features which causes a loss of spatial details especially along the object boundary.

The works discussed in this section so far are two-stage approaches, based on the underlying detection pipeline. The first stage produces ROIs and the second stage processes them independently to predict the class and mask. On the other hand, the one-stage approaches [35], [38], [44] bypass the stage of ROI generation. Instead, they simultaneously handle the bounding box prediction and classification, directly on the feature map of the whole image. It avoids the computational overhead of region-wise processing in the second stage of the two-stage approach. And as a result, it does not require pooling.

BlitzNet [16] is a one-stage network. It adopted a fully convolutional architecture and performed image-wise computation for simultaneous detection and segmentation. The accuracy of the network benefited from both tasks. But it did not make explicit use of the detection result to improve segmentation accuracy.

In general, one-stage methods in detection and instance segmentation trade off accuracy for speed compared to two-stage methods. However, it was recently pointed out that one-stage methods were also able to achieve state-of-the-art accuracy if the class imbalance of the bounding box proposals was addressed, for example by the "focal loss" function [35].

As mentioned earlier in this chapter, full-image based semantic segmentation methods have issues with small objects. But the result of instance segmentation shows accurate segmentation mask even for small objects. It indicates that the localization capability introduced by the object detection module helps with accurate pixel-wise classification. We suspect that object detection forces the network to maintain enough spatial information in its learned representations.

There have been some attempts in using object detection as an intermediate step to train an algorithm to extract features for semantic segmentation tasks [12], [13] rather than as the end result. Prior to deep learning methods, Pablo et al. [2] proposed a method where low-level object detectors were used to provide the pixel-wise features for semantic segmentation. They first combined low-level object detectors with high-level information to generate free-form region proposals and assigned scores to them that indicated the probability of belonging to an object category. Then these regions and their scores were used to generate pixel-wise features for training a number of SVM classifiers for semantic segmentation. BoxSup [12] explored using bounding box annotation in a weakly-supervised or unsupervised setting. Unsupervised region proposal methods [3], [30], [51] were applied to generate the candidate segmentation masks. The ground truth bounding box was used to pick the best segmentation mask from the candidates. This mask was used as the label to train the segmentation FCN and then got refined by the output of the FCN. The same procedure iterated to further train the FCN and refine the masks. They achieved competitive segmentation accuracy by only using a small number of segmentation labels and a large number of bounding box labels. But this approach was not end-to-end. The bounding box annotations did not directly supervise the training of the network. It was combined with external region proposal algorithms which generated the candidate segmentation masks. The accuracy of the network was bottlenecked by that of the region proposal algorithm. Jifeng *et al.* [13] used foreground segmentation masks generated by selective search [51] to locate the objects on the feature maps of the full image. The masks are then classified individually by feeding their features to fully connected layers. Instead of generating bounding boxes as a first step, they generated segmentation masks (super-pixels). However, like in [12], they also relied on external region proposal methods such as Selective Search [51]. In video segmentation tasks, object detection can be leveraged in a weakly-labeled setting where only image-level labels are available. Yu et al. [56] used an object detector to generate proposals for bounding boxes and free-form regions. These noisy proposals then got filtered based on their spatio-temporal consistency. The segmentation was further refined for each possible sequence of the object using shape likelihoods, which optimized object-level statistics, leading to better long-range consistency on the object-level.

2.5 Connection of Our Proposed Method to the Literature

Our proposed method on semantic image segmentation has drawn inspirations from the literature of object detection and instance segmentation. It is similar to Mask R-CNN in that they both incorporate the region proposal network to localize the object of interests. Both methods are the same as Faster R-CNN in terms of the region proposal generation. They differ in how the regions and the convolutional features are used in the downstream network. Mask R-CNN uses the ROI-Align layer to aggregate the features for each ROI and process them one-by-one with fully connected layers. Our method processes all the ROIs in one pass by introducing an ROI convolutional layer that allows direct convolution inside the ROIs. Chapter 3 will discuss our method in details.

Chapter 3

Method

This chapter starts off by further motivating our work by highlighting the challenges faced by neural networks in semantic segmentation, followed by our proposed method in details.

The networks for object classification are often used as the initial backbone architecture for semantic segmentation. These networks are re-purposed and fine-tuned for segmentation task. The problem is that some characteristics in the representation learned by the network are fundamentally different between classification and segmentation. Classification network is designed to be translation invariant. The idea is that the final prediction should be invariant to the locations of an object in the image space. On the other hand, semantic segmentation requires pixel-level accuracy. Classification network makes extensive use of pooling to build in the translation invariance while pooling is ideally to be avoided in the context of segmentation. To improve the segmentation accuracy, we need some mechanisms to compensate for the loss of spatial information due to pooling. Another problem in semantic segmentation that we aim to address is the difficulty in obtaining high-quality pixel-wise labels in practice. In contrast, it is easier to obtain labels for object detection: bounding boxes locations and their classes. It was reported that the semantic segmentation labels took more that 15 times longer to finish than spotting the instances [12] when annotating MS COCO dataset. If we could incorporate the available object detection labels in the training of semantic segmentation task, it may help reduce the need for segmentation labels. Note that we use bounding box and ROI interchangeably throughout the text.

In this work, we develop a new mechanism that leverages the ideas in object detection to improve the localization capability of the segmentation network. Instead of viewing object detection as part of the final objective as in instance segmentation, we propose to use it as an intermediate step to guide the training of the network, such that the network maintains spatial information in the early stages of its learned representations. We show later that this approach helps to improve the segmentation accuracy. Because object detection is integrated in the network, it also naturally addresses the second issue discussed above about the need for labels. With the new method, we can directly train the network with bounding box labels.

The overall architecture is shown in Figure 3.1. We essentially break down the segmentation task into two stages: first localizing a rough region in which the object has been detected, then performing segmentation in the region. The first part of the network is Region-Proposal-Network (RPN) (Section 3.2) which is part of the backbone of Faster R-CNN. The bounding box prediction from RPN is used as an extra input to the downstream convolutional layers for segmentation (Section 3.4) to improve its localization capability. We introduce an ROI convolutional layer (Section 3.3) that directly convolves on all the ROIs in one pass without the need of ROI pooling or other region-wise processing, and show that this layer is trainable through backpropagation. The proposed network can be trained end-to-end.

3.1 Backbone

VGG16 [49] and ResNet [23] are two of the most popular choices for backbone feature extractors. We use VGG16 as the feature extractor, i.e., the backbone of the architecture, since it is a simpler network and do not need as much data to train. It is shown in Fig. 2.1. The network that we propose here could be built on top of a deeper backbone like ResNet [23] or FPN [34] to further improve accuracy. But a simpler backbone helps to keep the focus on showing that adding a localization module improves the segmentation accuracy

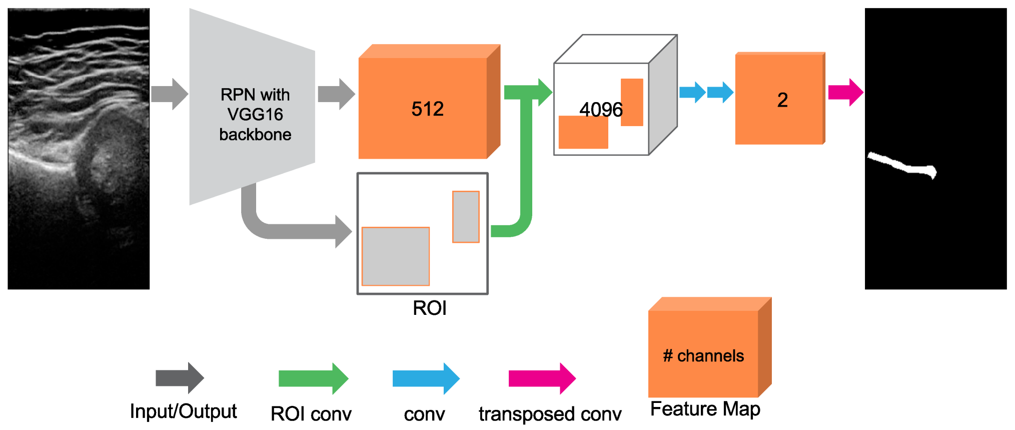


Figure 3.1: Overall architecture of our network. The legend at the bottom illustrates different components. Best viewed in color. The input to the network can be any 2D grayscale (duplicated to 3 channels) or color image. The RPN network is built on top of VGG16. It takes in the input image and outputs feature maps of 512 channels and a number of ROIs. The arrow for 'ROI conv' represents one ROI convolutional layer that takes as input the feature map and the ROIs. Each 'conv' and 'transposed conv' arrow represents one convolutional layer and transposed convolutional layer, respectively.

compared to vanilla segmentation networks (see results in Chapter 4).

3.2 Region Proposal Network

We adopt the Region Proposal Network (RPN) from Faster R-CNN [45] to perform object localization. Each region is classified as foreground or background. The foreground regions are passed to the downstream segmentation network where each pixel inside the regions is classified. The pixels outside these regions are considered as background. The architecture of the RPN is shown in Fig. 3.2. It applies convolutional layers on top of the features extracted by the backbone feature extractor, which is VGG16 in this case.

The network is trained to optimize two losses simultaneously: a classification loss on the class of the bounding box, and a regression loss on the location of the bounding box. We follow the formulation in Faster RCNN: consider a binary classification loss, that is, the bounding box either belongs to the foreground or background. We rewrite the loss function from [45] here in Eq 3.1 for completeness:

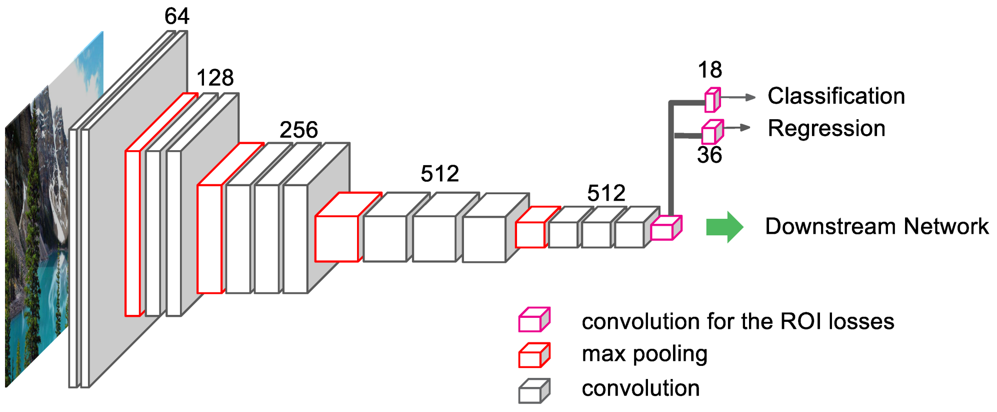


Figure 3.2: RPN with VGG16 backbone

$$L_{rpn} = \frac{1}{\lambda_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \frac{1}{\lambda_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{3.1}$$

where i is the index of the anchor box defined on the image. λ_{cls}, λ_{reg} are the normalization coefficients for classification loss L_{cls} and regression loss L_{reg}. p_i^* and p_i are the ground truth and predicted class probability for the positive class, respectively. t_i^* and t_i are the ground truth and predicted bounding box parameters, respectively. The regression loss is multiplied with the true class so that the negative class is ignored. In practice, L_{cls} and L_{reg} are both normalized by the mini-batch size of the ROIs: $\lambda_{cls} = \lambda_{reg} =$ ROI batch size. The negative samples of ROIs do not contribute to the regression loss. The bounding boxes are parameterized the same way as in [45] and [19], with

$$\begin{aligned}
t_x &= (x - x_a)/w_a, & t_y &= (y - y_a)/h_a, \\
t_w &= log(w/w_a), & t_h &= log(h/h_a), \\
t_x^* &= (x^* - x_a)/w_a, & t_y^* &= (y^* - y_a)/h_a, \\
t_w^* &= log(w^*/w_a), & t_h^* &= log(h^*/h_a)
\end{aligned} \tag{3.2}$$

where x,y,w,h denote the center coordinates and the width and height of the predicted bounding box. x^*,y^*,w^*,h^* denote those of the ground truth bounding box. w_a and h_a denotes the width and height of the anchors. Each bounding box is represented as a 4-tuple: $t_i = (t_x,t_y,t_w,t_h)$, which are relative to the location and size of the anchors. They can be interpreted as scale-invariant translation and log-space shift of width and height relative to the anchors. Again, the * denotes a variable as the ground truth.

The classification loss is a binary cross entropy loss: $L_{cls}(p_i, p_i^*) = -p_i^* \, log(p_i)$. The regression loss is a smooth L1 loss, also called a Huber Loss [20]: $L_{reg}(t_i, t_i^*) = \sum L_{huber}(t_i, t_i^*)$.

$$L_{huber}(x) = \begin{cases} 0.5\sigma^2 x^2, & \text{if } |x| \leq \frac{1}{\sigma^2} \\ |x| - \frac{0.5}{\sigma^2}, & \text{otherwise} \end{cases} \tag{3.3}$$

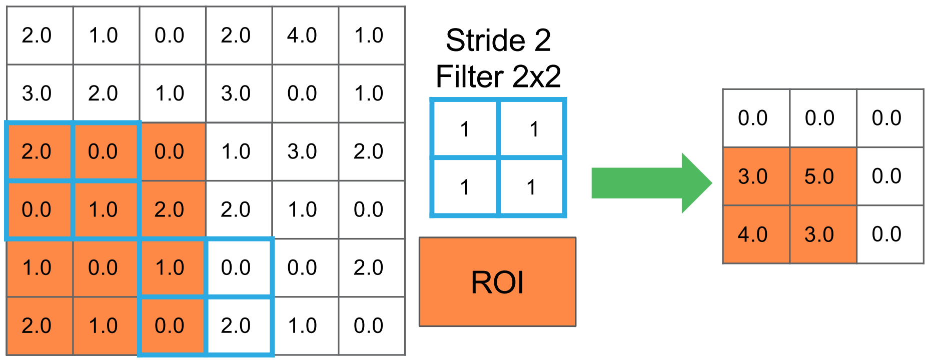
The σ is a parameter that controls the boundary of the L2 loss and L1 loss. It is set to 3 throughout the experiments in Faster R-CNN and our experiments.

3.3 ROI Convolution

The output of the RPN is a list of ROIs in terms of bounding boxes. The features for the ROIs can be extracted from the feature map of the previous layer and passed to downstream segmentation network. Region-wise method has been widely used [13], [22], [45] where each ROI is processed individually as a new input image. To obtain a fixed-size input, each ROI is first fed into a pooling layer which produces a fixed-size grid. The features for each cell in the grid are those of the pixel that has the maximum activation in that cell.

In contrast, we propose an image-wise method to simplify the computation, where the downstream segmentation network processes all the ROIs in one shot. In order to do that, a new layer called ROI convolution is introduced. A similar idea has been proposed in [7]. But they used ROI convolution only in the inference time to speed up computation. We show that ROI convolution is differentiable and is trainable through backward propagation. We use ROI convolution at both training and inference time.

The schematic diagram of ROI convolution is illustrated in Fig. 3.3a. For comparison we also include an illustration for ROI pooling in Fig. 3.3b. Note the edge case in the ROI convolution, where we want to compute the output feature for a pixel at the boundary of the ROI. The filter covers some pixels outside the ROI. Instead of masking out these pixels, we use the original features of them during the computation as long as the target pixel is inside



For the pixels at the edges of the ROI, the features outside the ROI are not masked out during the convolution as long as the the top-left pixel covered by the filter is inside the ROI.

(a) ROI convolution

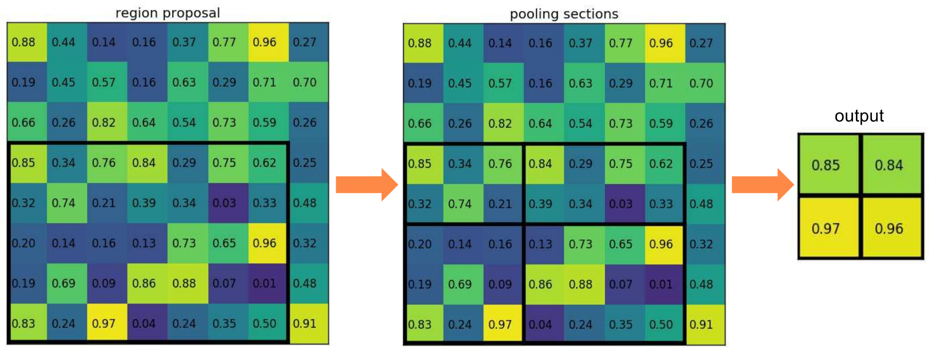


Image Credit:
`https://blog.deepsense.ai/region-of-interest-pooling-explained`

(b) ROI Pooling

Figure 3.3: Comparison of operators on ROI

the ROI. The idea is that we want to account for the context of the target pixel during the convolution.

We compare the architecture of FCN-32s [39], Faster R-CNN [45], and our network RPN-FCN in Fig. 3.4, highlighting the key difference in how the features of the ROIs are extracted and passed to the downstream network. All the three networks use VGG16 as the backbone. They differ from the step following the ‘conv5’ feature map of VGG16. Faster R-CNN applies ROI pooling to transform to region-wise computation on each ROI. FCN applies standard max pooling on the full feature map and produces a full-size probability map

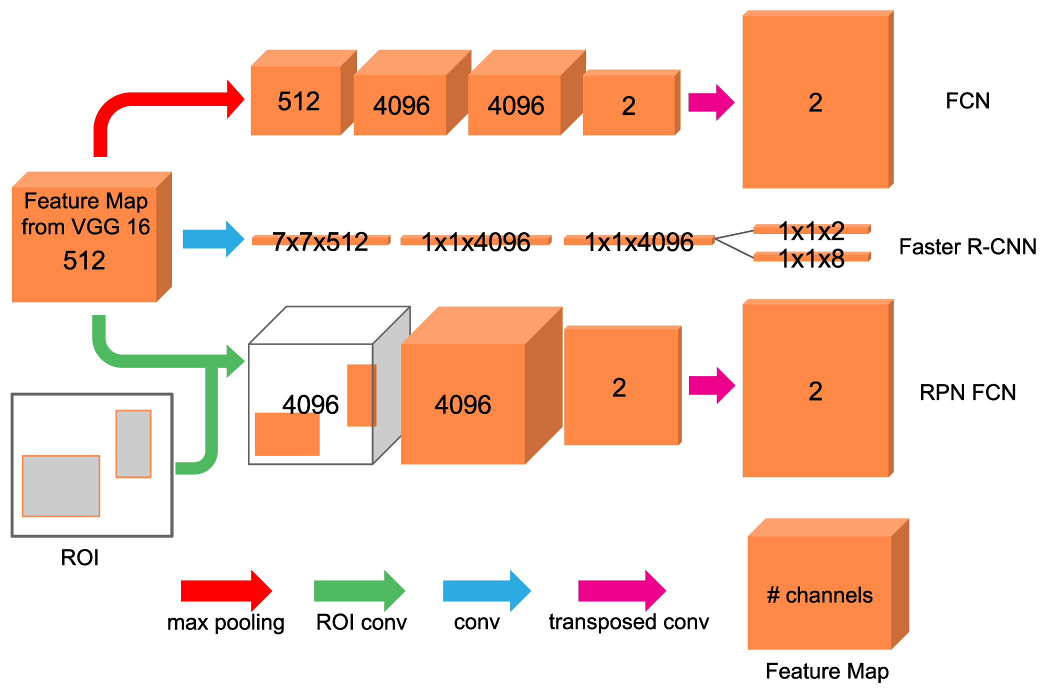


Figure 3.4: Comparison of network architectures of FCN-32s, Faster R-CNN and RPN-FCN

in the end. In RPN-FCN, we use our proposed ROI convolution, where the convolution is applied only within the ROIs. This operation is performed image-wise, which means that the output will be the feature map for the full image instead of for each ROI. Note that the networks illustrated in the figure are based on a two-class scenario. FCN and RPN-FCN both perform segmentation, and Faster R-CNN does detection. They are not directly comparable due to their different goals but it is meant to show the difference in their use of the VGG-16 feature map.

3.3.1 Forward Pass of ROI Convolution

The forward pass of ROI convolution is shown in Eq. 3.4.

$$y^l_{i,j} = (w^l * z^{l-1})_{i,j} + B^l = (\mathbb{1}[(i,j) \in \cup\text{ROI}] \sum_{a=0}^{k-1} \sum_{b=0}^{k-1} w^l_{a,b} z^{l-1}_{i+a,j+b}) + B^l \quad (3.4)$$

where $y^l_{i,j}$ represents the pre-activation output of layer l at pixel (i,j), at one of the channels. z^l represents the activation of layer l. w^l is the filter at layer

l, of size $k \times k$. B^l is the bias at layer l. In practice, it is usually a scalar for each channel. $*$ denotes the ROI convolution. $\cup$ROI denotes the union of the set of indices inside all the ROIs. The indicator function checks if the pixel (i, j) is inside any ROI. The forward pass is just like the regular convolution except that the pixels outside the ROIs are set to zeros.

3.3.2 Backpropagation of ROI Convolution

The backpropagation of ROI convolution can be computed as follows. Suppose that we have the cost function E, and $\frac{\partial E}{\partial z^l} = \delta^l$ for the current layer l, we need to compute $\frac{\partial E}{\partial z^{l-1}}$, $\frac{\partial E}{\partial w^l}$ and $\frac{\partial E}{\partial B^l}$:

$$\frac{\partial E}{\partial z^{l-1}_{i,j}} = \sum_{a=0}^{k-1}\sum_{b=0}^{k-1} \delta^l_{i-a,j-b}\sigma'_{i-a,j-b} w^l_{a,b} \mathbb{1}[(i-a, j-b) \in \cup\text{ROI}] \tag{3.5}$$

$$\frac{\partial E}{\partial w^l_{a,b}} = \sum_{j=0}^{W^l_o}\sum_{i=0}^{H^l_o} \delta^l_{i,j}\sigma'_{i,j} z^{l-1}_{i+a,j+b} \mathbb{1}[(i, j) \in \cup\text{ROI}] \tag{3.6}$$

$$\frac{\partial E}{\partial B^l} = \sum_{j=0}^{W^l_o}\sum_{i=0}^{H^l_o} \delta^l_{i,j}\sigma'_{i,j} \tag{3.7}$$

where W^l_o, H^l_o denotes the width and height of the output of layer l. $\sigma'_{i,j}$ denotes the derivative of the activation function at pixel (i,j). Unlike in regular convolution, the pixels outside the ROIs do not contribute to the gradient in the ROI convolutional layer. We give the derivation of the backpropagation equations below. For regular convolution, we need to sum up the gradients of all the pixels on the output feature map:

$$\begin{aligned}\frac{\partial E}{\partial w^l_{a,b}} &= \sum_{j=0}^{W^l_o}\sum_{i=0}^{H^l_o} \frac{\partial E}{\partial z^l_{i,j}} \frac{\partial z^l_{i,j}}{\partial y^l_{i,j}} \frac{\partial y^l_{i,j}}{\partial w^l_{a,b}} \\ &= \sum_{j=0}^{W^l_o}\sum_{i=0}^{H^l_o} \delta^l_{i,j}\sigma'_{i,j} \frac{\partial(\sum_{b=0}^{k-1}\sum_{a=0}^{k-1} w^l_{a,b} z^{l-1}_{i+a,j+b} + B^l)}{\partial w^l_{a,b}} \\ &= \sum_{j=0}^{W^l_o}\sum_{i=0}^{H^l_o} \delta^l_{i,j}\sigma'_{i,j} z^{l-1}_{i+a,j+b} \end{aligned} \tag{3.8}$$

$$\begin{aligned}
\frac{\partial E}{\partial B^l} &= \sum_{j=0}^{W_o^l}\sum_{i=0}^{H_o^l} \frac{\partial E}{\partial z_{i,j}^l}\frac{\partial z_{i,j}^l}{\partial y_{i,j}^l}\frac{\partial y_{i,j}^l}{\partial B^l} \\
&= \sum_{j=0}^{W_o^l}\sum_{i=0}^{H_o^l} \delta_{i,j}^l \sigma'_{i,j} \frac{\partial(\sum_{b=0}^{k-1}\sum_{a=0}^{k-1} w_{a,b}^l z_{i+a,j+b}^{l-1} + B^l)}{\partial B^l} \\
&= \sum_{j=0}^{W_o^l}\sum_{i=0}^{H_o^l} \delta_{i,j}^l \sigma'_{i,j} \qquad (3.9)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial E}{\partial z_{i,j}^{l-1}} &= \sum_m\sum_n \frac{\partial E}{\partial z_{m,n}^l}\frac{\partial z_{m,n}^l}{\partial y_{m,n}^l}\frac{\partial y_{m,n}^l}{\partial z_{m,n}^{l-1}} \\
&= \sum_m\sum_n \delta_{m,n}^l \sigma'_{m,n} \frac{\partial(\sum_{b=0}^{k-1}\sum_{a=0}^{k-1} w_{a,b}^l z_{m+a,n+b}^{l-1} + B^l)}{\partial z_{i,j}^{l-1}} \qquad (3.10) \\
&= \sum_{i-a}\sum_{j-b} \delta_{i-a,j-b}^l \sigma'_{i-a,j-b} w_{a,b}^l \qquad (3.11) \\
&= \sum_{a=0}^{k-1}\sum_{b=0}^{k-1} \delta_{i-a,j-b}^l \sigma'_{i-a,j-b} w_{a,b}^l \qquad (3.12)
\end{aligned}$$

The transformation from Eq. 3.10 to Eq. 3.11 is based on the following: the partial derivative will be zero except for $i = m + a$, $j = n + b$, reorder the terms: $m = i - a$, $n = j - b$, the equation can therefore be re-written by substituting the subscripts.

From these backpropagation equations 3.8, 3.9 and 3.12, it is trivial to derive the backpropagation for ROI convolution. The idea is that the features at layer $l-1$ are not only masked by the ROI during the forward computation, but they are also masked during the gradient computation.

3.4 Transposed Convolution

Transposed convolution is a standard operation in FCN network for upsampling the feature map from a small size into a full size of the input image. It has got its name from that it can be thought of a regular convolution with a transposed filter f^T.

Fig. 3.5 is an illustration of a transposed convolution on an input image of size 2 x 2, that has a filter 3 x 3, stride 2, padding 2 [15]. In practice, a

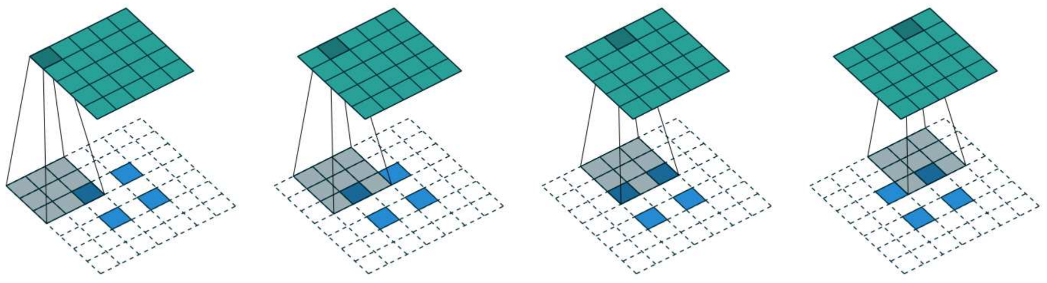


Figure 3.5: Transposed convolution of filter 3 x 3, input 2 x 2, stride 2, padding 2 [15].

transposed convolution is implemented by swapping the forward and backward function of a regular convolutional layer.

3.5 Loss Function

We use a multi-task loss that is similar to that defined in Mask RCNN [22]: $L = L_{rpn} + L_{seg}$ where the loss for RPN is combined with the segmentation loss L_{seg}. Different from the loss in [22], our loss is defined over the full image as opposed to each ROI. The loss for RPN, $L_{reg} + L_{cls}$, is the average loss for all ROIs, same as in [45]. The segmentation loss L_{seg} is the standard cross-entropy loss for the entire image:

$$L_{seg} = -\sum_{i} \sum_{c=0}^{k} log\, p_i(y = c) \tag{3.13}$$

where i is the index of the pixel on the output probability map. c is the index of class, in the set {0,2,...,k}. $p_i(y = c)$ is the predicted probability of pixel i belonging to class c. This multi-task loss is back-propagated through all the layers in the network including both the segmentation layers and RPN.

3.6 Implementation of Convolution and ROI Convolution

3.6.1 Caffe's Implementation of Convolutional Layer

Convolution can be thought of as sliding the filter across the input feature map, inserting the dot product of the filter and the input patch at that location into the output at the location of the center pixel. To avoid a for-loop for computing the dot product at each location which would be slow, Caffe approaches convolution as a multiplication of two big matrices to speed up the computation. The core idea is that all the patches across the input image are reshaped and concatenated into one matrix. The filter is reshaped, replicated and organized into another matrix. One can then use an optimized linear algebra library to perform the matrix multiplication. This process is illustrated in Fig.3.6.

3.6.2 Implementation of ROI Convolutional Layer

We implemented the ROI convolutional layer in Caffe. The logic is very similar to the convolutional layer except that the reshaped data need to account for the ROI. In particular, in the forward pass, if a pixel is outside the ROIs, its surrounding patch covered by the convolutional filter will be set to zeros in the reshaped feature map D shown in Fig. 3.6. In the backward pass, based on Equation 3.5 and Equation 3.6, the gradient w.r.t. the input features z and the gradient w.r.t. the filters w are both computed based on the gradient w.r.t. the features in the next layer, which is set to zero for any pixel outside the ROIs.

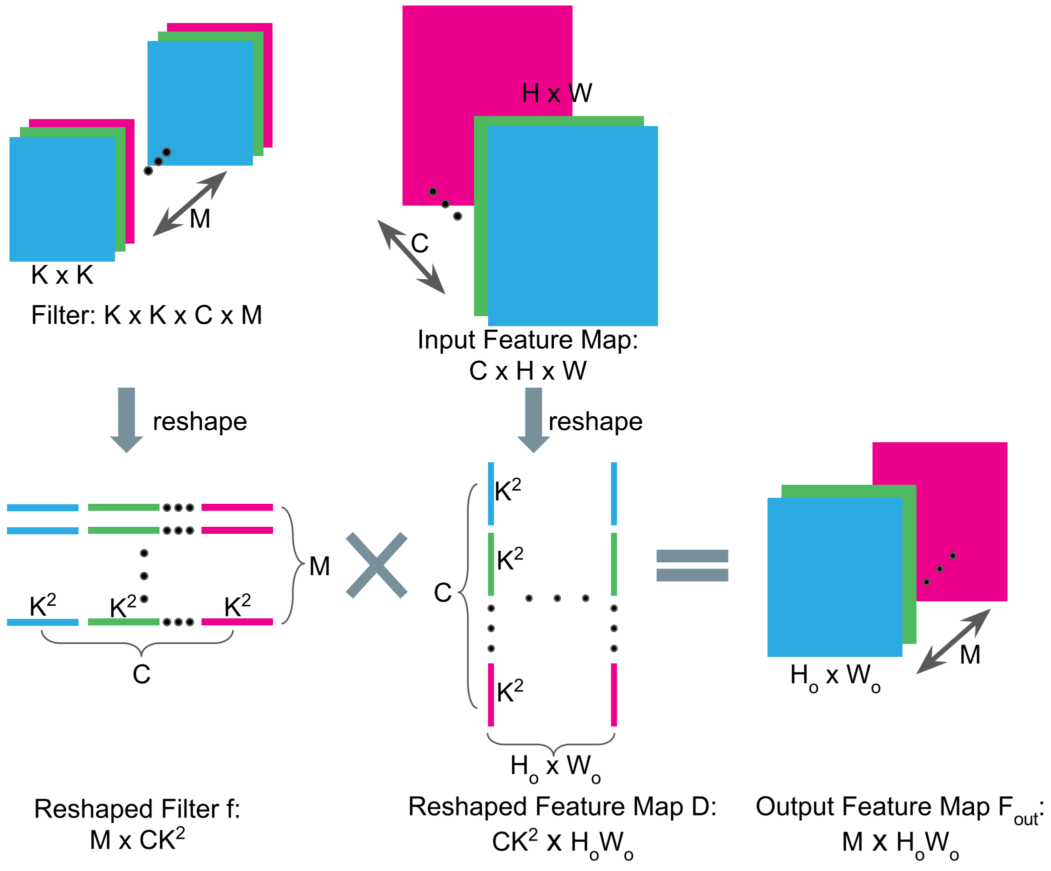


Figure 3.6: The implementation of convolution in Caffe

Chapter 4

Experiments

4.1 Dataset

4.1.1 Hip Ultrasound Data

Experiments were performed on a clinical dataset of 3D ultrasound (US) images of infant hips for diagnosis of developmental dysplasia of the hip (DDH). DDH, a condition affecting up to 1 in 30 infants, is easy to treat if diagnosed early, but if undiagnosed can result in early onset osteoarthritis, and accounts for around 30 of all hip replacements in patients under 60. Current DDH's diagnosis based on 2D Ultrasound is notoriously operator-dependent, requiring years of training to reach reasonable and more importantly reproducible performance [25].

The segmentation of acetabulum bone is an essential step towards making diagnosis of DDH fully automated and less reliant on the operator. However the task is very challenging due to the highly noisy nature of ultrasound images compared to CT or MRI images, and the shadowing/discontinuities present in the boundaries of the bone. And the classes are highly imbalanced: the acetabulum only takes up a small portion of the image (0.3 percent in our dataset).

Data collection was performed with institutional Health Ethics Review Board approval and written informed consent from parents, at the Radiology Department at University of Alberta. Three-dimensional ultrasound scans of infants' hip, were obtained on a Philips iU22 scanner (Philips Healthcare,

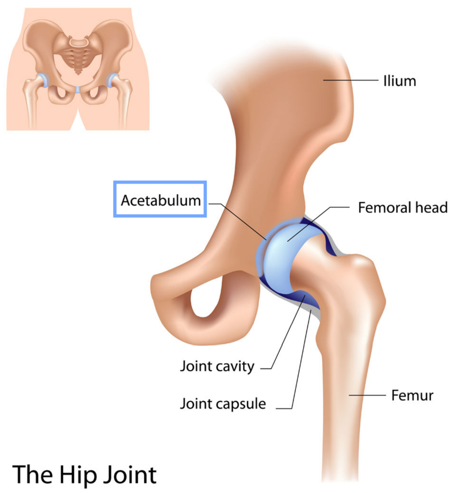
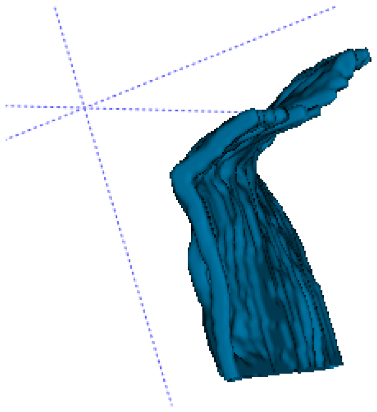
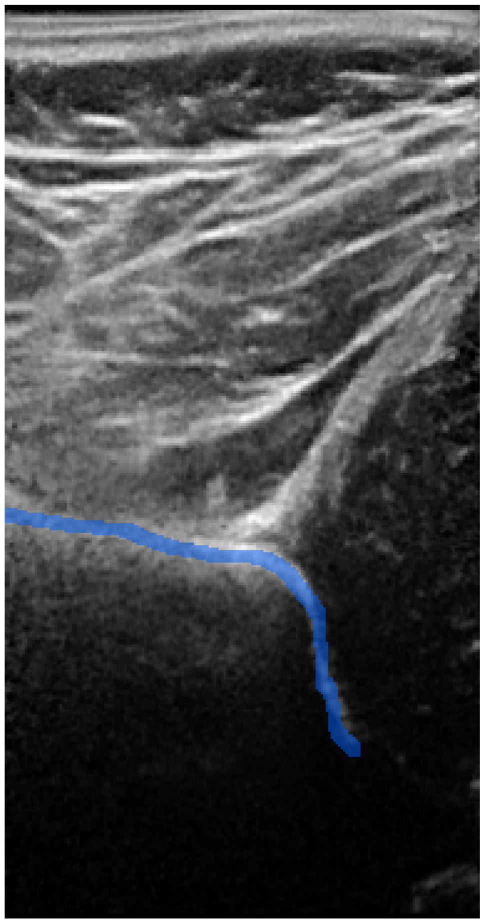


Image Credit: `https://goo.gl/JgmtZu`

(a) Anatomy of hip



(b) 3D view of acetabulum



(c) Ultrasound image. Blue mask is overlayed on top of the acetabulum

Figure 4.1: Hip anatomy and the ultrasound

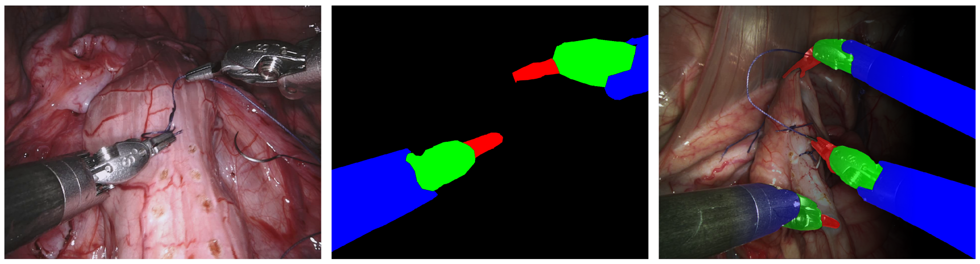
Andover, Mass) using a 13 MHz linear (Philips 13VL5) transducer in coronal orientation, harmonics 'off'. The transducer was steered over an angular range of $\pm 15°$in 3.2 seconds.

The anatomy of the hip and its ultrasound image are illustrated in Fig. 4.1. Each 3D ultrasound volume is comprised of 256 ultrasound slices of 0.13 mm thickness containing 411×192 pixels measuring 0.11×0.20 mm. The volumes were exported to Cartesian DICOM format and used for segmentation.

The dataset consists of 49 three-dimensional ultrasound scans that has manually annotated segmentation labels for acetabulum provided by the lead pediatric musculoskeletal radiologist. We randomly divide them into a training set of 30 scans and a testing set of 19 scans. The scans from the same patient either all go into the training or testing set. We crop some of the scans to make all the 2D slices share a unified size of 367×192 pixels (height by width). Most 3D volumes contain 256 slices.

4.1.2 Robotic Instrument Dataset - Endoscopic Images Vision Challenge 2017

Robotic Instrument Dataset is a public dataset released as part of the Endoscopic Images Vision Challenge 2017 [46] at the Medical Image Computing and Computer Assisted Interventions Conference (MICCAI) 2017. An example of the data is shown in Fig. 4.2. The dataset contains 10 videos each with 300 images, captured at 2 Hz. Out of these 10 videos, two full videos and the last 75 frames of each of the other eight videos are used as a hold-out set for testing. The rest are the training set: the first 225 frames of each of the eight videos. Each frame is an RGB image. This challenge consisted of three tasks: a binary segmentation task and two multi-class segmentation tasks. In our experiment, we focus on the binary segmentation task only.



(a) Raw Image (b) Image Label (c) Overlay

Image Credit: `https://endovissub2017-roboticinstrumentsegmentation.grand-challenge.org`

Figure 4.2: Robotic instrument data

4.2 Evaluation Metrics

We evaluated the segmentation using precision, recall and dice score on 2D slices, with

$$\begin{aligned} Precision &= \frac{TP}{TP + FP} \\ Recall &= \frac{TP}{TP + FN} \\ Dice\ Score &= \frac{2TP}{2TP + FP + FN} \end{aligned} \tag{4.1}$$

where TP, FP, FN denotes true positive, false positive, and false positive, respectively. Note that dice score is the same as F1 score. We did not include Intersection Over Union (IOU, or equivalently, Jaccard Coefficients), since dice score and IOU follow the same trend (if one is higher, the other must be higher too), and dice score is more common in medical imaging literature.

4.3 Implementation Details

Data Pre-processing All the data are mean-subtracted, following the convention of FCN and Faster R-CNN. We augmented the training data of the the hip ultrasound data by horizontal flipping. We do not perform post-processing.

Network structure We used the same VGG16-based RPN as the one in the original Faster R-CNN [45], except that we changed the padding of the first convolutional layer to 50 to account for the reduction of the image size, following the convention of FCN. The transposed convolutional layer was modified from the original FCN-32s network [39], where the number of output channels was reduced from 21 to 2 for binary segmentation.

Solver hyper-parameters The network was trained using SGD with a momentum of 0.9 and weight decay of 0.0005. Learning rate was set to $1e^{-5}$ for the hip ultrasound data and $1e^{-4}$ for the robotic instrument data, and reduced by a ratio of 0.1 for every 50k iterations. The segmentation is normalized by the number of pixels on the image to balance the losses of object localization and segmentation. During training, a dropout of 0.5 was used in the two convolutional layers following RPN. If not specifically mentioned, for the network layers that were the same as in FCN and Faster R-CNN, we tried to keep the parameters the same as those used in the original work to make a fair comparison. The implementation was based on Caffe [27], where we added the implementation of ROI convolutional layer. The experiment was carried out on an NVIDIA Pascal 1080 GPU.

4.4 Results and Discussion

The proposed network was designed with small object segmentation in mind, i.e., imbalanced data. In this section, we first show how it performs on the acetabulum segmentation where the data is imbalanced. We then present the result on the robotic instrument segmentation where the data is balanced. We conduct ablation studies on the acetabulum segmentation to investigate the impact of ROI convolutional layer in training and inference time.

4.4.1 Acetabulum Segmentation on Hip Ultrasound

We compared our method with its FCN counterpart FCN-32s, and U-Net [47]. We used the original Caffe implementation for FCN-32s. For U-Net, we tried three different implementations: the original Caffe implementation, an implementation included in the 3D-U-Net [10], and a Tensorflow [41] implementation. The best result was reported, which was achieved with the code base of 3D-U-Net. Note that 3D-U-Net comes with an on-the-fly deformation layer that augment the training data at every iteration. It may be the reason why it performed better than the other implementations. For FCN-32s and

Table 4.1: Comparison of our proposed method **RPN-FCN** with FCN-32s and U-Net. Scores are averaged over all the test slices. **Bold** numbers denote the best of these three methods. For all the methods, the best results from hyper-parameter tuning and different implementations are reported.

Method	Precision	Recall	Dice Score
RPN-FCN	**31.29**	**40.16**	**35.18**
FCN-32s	18.24	28.54	22.25
U-Net	3.94	6.33	4.86

RPN-FCN, we only augmented the data with horizontal flipping.

The test scores are summarized in Table 4.1. The results that we reported were the mean scores for all the 4864 test slices. We did not cherry-pick test slices. The distribution of the dice score were shown in Fig. 4.3 by sorting the slices that contain positive labels based on their dice score. Our method RPN-FCN achieves better overall segmentation performance than the other two methods. And it yields an improved dice score on a wide range of slices. Adding RPN on top of FCN-32s constrains the segmentation problem in a smaller region. The result suggests that it is an effective way to train the downstream segmentation layers. Outperforming U-Net also shows the efficacy of this localization mechanism.

Our network is similar to FCN-32s in that the backbone are both VGG16 and the upsampling paths are similar. The major difference lies in the use of ROI convolution after 'conv5' layer of VGG16. Due to the removal of the pooling layer, the feature map has a higher resolution. It only needs to be upsampled by a factor of 16 instead of 32 as in the original FCN. Upsampling from a higher resolution feature helps to retain the features of small objects, therefore improving the segmentation accuracy.

The dice score is low compared to other medical image segmentation tasks. The main reason is that the object of interest is small. Full-image based networks generally have some issues dealing with this type of imbalanced data [29]. The representations learned by the network are dominated by the majority class, in this case, the background. U-Net introduced class weighting in

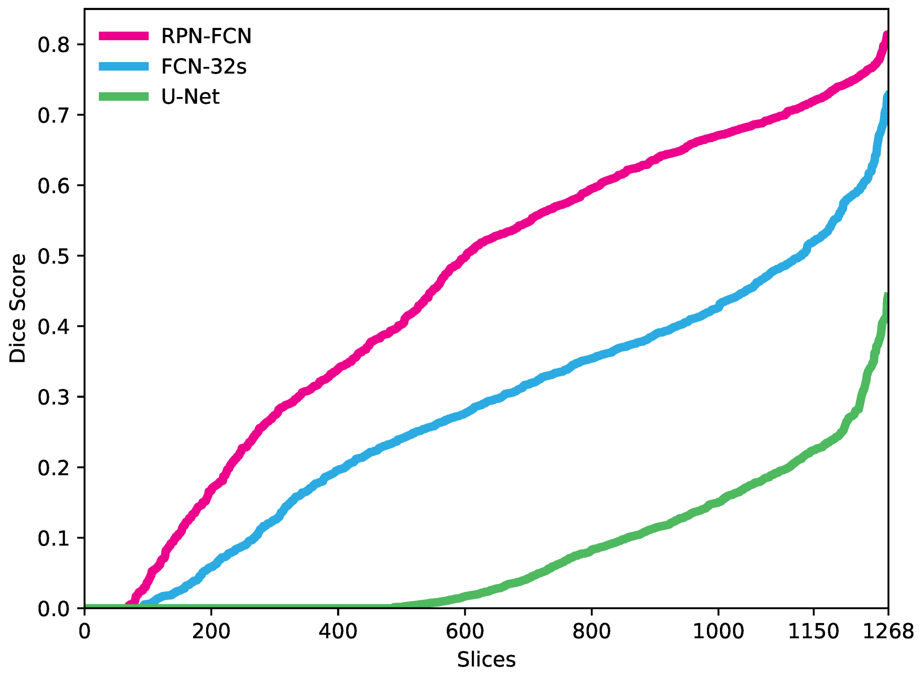


Figure 4.3: Dice score of all the test slices that contain positive labels (1268 out of the total of 4864), sorted in ascending order of dice score.

the loss to combat this issue. We tried weighting by the class frequency but did not see improvement. The best result for U-Net was in fact obtained from the 3D U-Net implementation of the 2D network, where the input was a 2D image patch instead of the full image. It might be possible to achieve better performance with U-Net by extensive parameter tuning on the class weights or the solver. But our main point here is that small object segmentation poses challenges on full-image-based methods. The detection mechanism in our proposed network offers an effective means for alleviating the issues in training the network.

We visualized the segmentation results of some slices in Fig. 4.4. Our network was able to produce smooth regions when the shape is relatively clear in the view. A few unsuccessful segmentations of our method were shown in Fig. 4.5. Figure 4.5a and Figure 4.5c both present a shape very similar to acetabulum which lead to some false positives. Figure 4.5b presents a shape with discontinuous intensity (the dark region between the two ends of the shape), leading to false negatives. These cases are generally hard for any

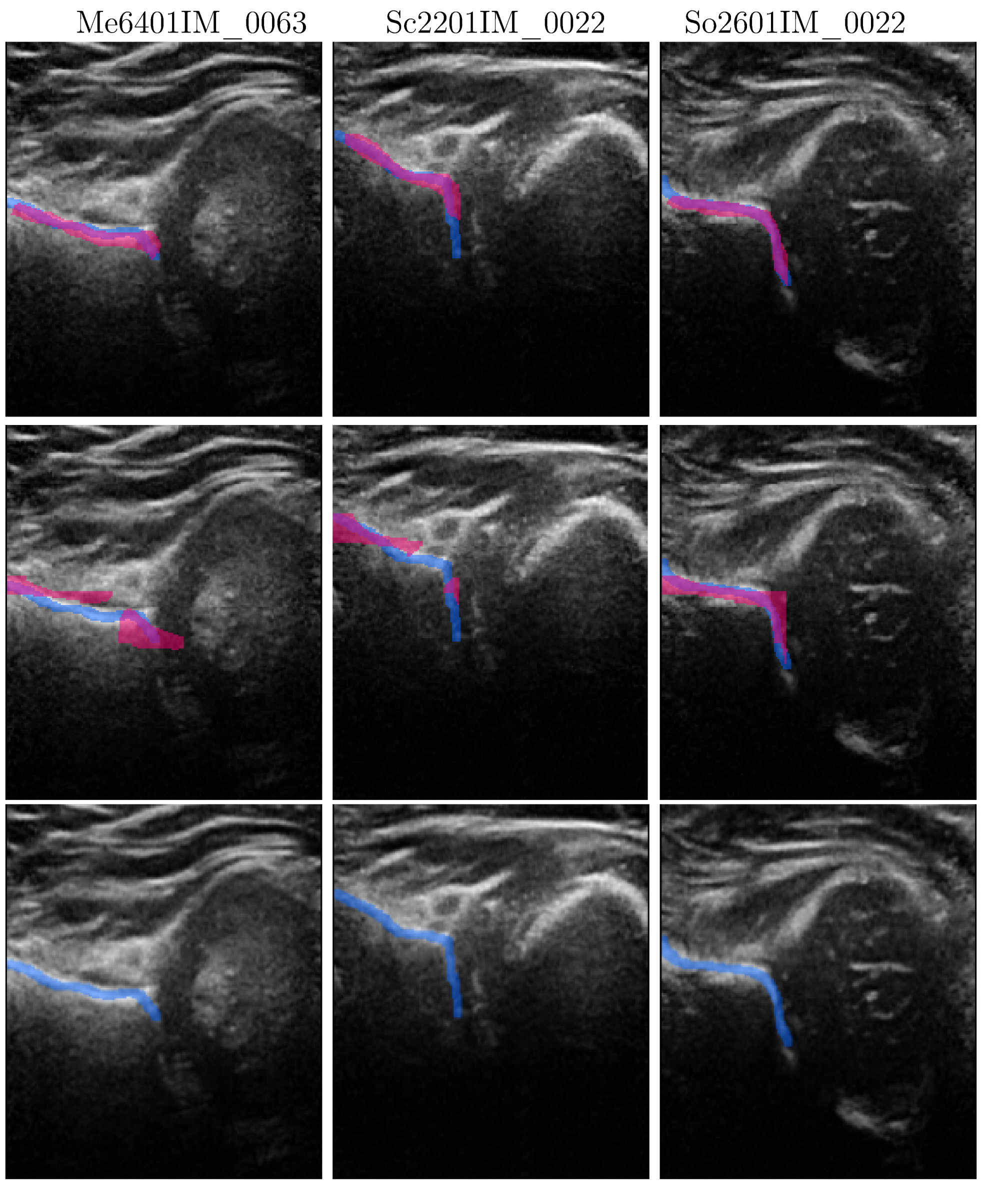


Figure 4.4: Segmentation result on slices from different volumes. Blue masks denote the ground truth; purple denotes the prediction. Images were cropped from top and bottom for a better view. The heading in each column is the name of the volume containing the slice. The three rows correspond to the results of our method RPN-FCN, FCN-32s, and U-Net, respectively. There is no visible prediction for U-Net since it predicted all pixels as background.

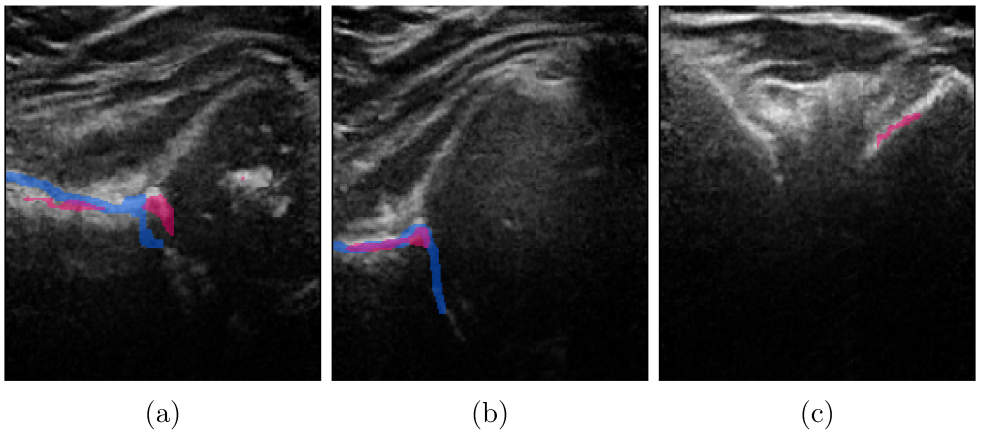


Figure 4.5: Failure cases. Blue masks denote the ground truth; purple denotes the prediction.

Table 4.2: Comparison of our proposed method **RPN-FCN** with FCN-32s on robotic instrument segmentation. Scores are averaged over all the test images. **Bold** numbers denote the best of these two methods.

Method	Precision	Recall	Dice Score
RPN-FCN	**91.71**	89.69	90.69
FCN-32s	91.32	**90.03**	90.67

method. This again explains the relatively low dice score.

4.4.2 Robotic Instrument Segmentation

As the test labels were not publicly released, we could not compare with the official test result of the MICCAI challenge.For this experimental study, we applied our proposed RPN-FCN on this dataset to understand how it compared with FCN, which was our participating entry in this challenge. The test result was based on the frames in video 8 where the labels were given. Both networks were trained with the training data in video 1 to 7. Video 8 was chosen as the test set here because it had a medium level of difficulty based on the average results of all the teams in the challenge.

Table 4.2 summarized the results. We did not see significant differences in the performance of the two methods. This was expected since the ROI convolution would be almost identical to a conventional convolution if the

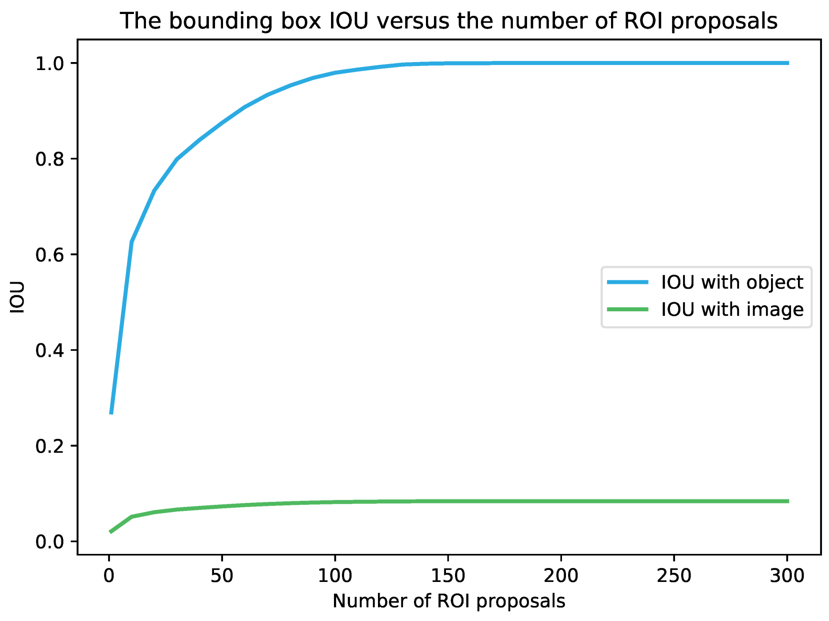


Figure 4.6: IOU versus the number of ROI proposals

bounding box of the object covers the majority of the image. It was typically the case in balanced data.

4.4.3 Ablation Experiments

To investigate the impact of ROI convolution, we performed ablation experiments on acetabulum segmentation task as follows.

IOU of the ROI proposals with the ground truth We analyzed how the number of ROI proposals affected the IOU with the ground truth bounding boxes. We varied the number of ROI proposals from 1 to 300 during the inference time. The average result on all the test slices with positive samples is illustrated in Fig. 4.6. In fact, our best result with ROI convolution was achieved with 300 ROI proposals during inference time. The segmentation accuracy goes down as the number of ROI proposals reduces. As we can see from the IOU curve, 300 ROIs cover the entire object without covering the full image. This makes the segmentation easier since it confines the search space in a local region around the object.

ROI convolution versus conventional convolution We first replaced

Table 4.3: Comparison of RPN-FCN with or without using the detection loss and ROI convolution in training, on acetabulum segmentation. Scores are averaged over all the test slices. **Bold** numbers denote the best of these methods.

Method	Precision	Recall	Dice Score
FCN-noPool5-scratch	20.40	25.88	22.81
RPN-FCN	**31.29**	**40.16**	**35.18**
FCN-32s	18.24	28.54	22.25

the ROI convolutional layer with conventional convolutional layer during inference time while loading the same weights. The segmentation result was identical, which we found to be a bit surprising. It showed that the network performed equally well even when the convolution is applied on the full-size of the image. In other words, the benefit of using ROI convolution was not at the inference time. We asked the question: did ROI convolution help in training time? Although it seemed to be true based on the dice score comparison with FCN-32s and U-Net, could the performance difference instead be related to the small changes that we made to the FCN-32s network architecture, such as smaller padding and removal of a pooling layer?

To answer this question, We conducted an experiment where we took the exact architecture of our RPN-FCN, removed the detection branch, replaced ROI convolution with conventional convolution, trained it from segmentation loss only. We named this method as "FCN-noPool5-scratch". The result is shown in Table 4.3. The segmentation results from RPN-FCN and FCN-32s are included for comparison. RPN-FCN performed significantly better than its counterpart "FCN-noPool5-scratch" trained from segmentation loss, while the dice scores from FCN-32s and "FCN-noPool5-scratch" were similar. This showed the benefit of training the network with the localization unit and the ROI convolution. On one hand, it allows the use of box-level labels for training. On the other hand, it inherently forced the network to learn features useful for localization and therefore helps improve the segmentation on small objects.

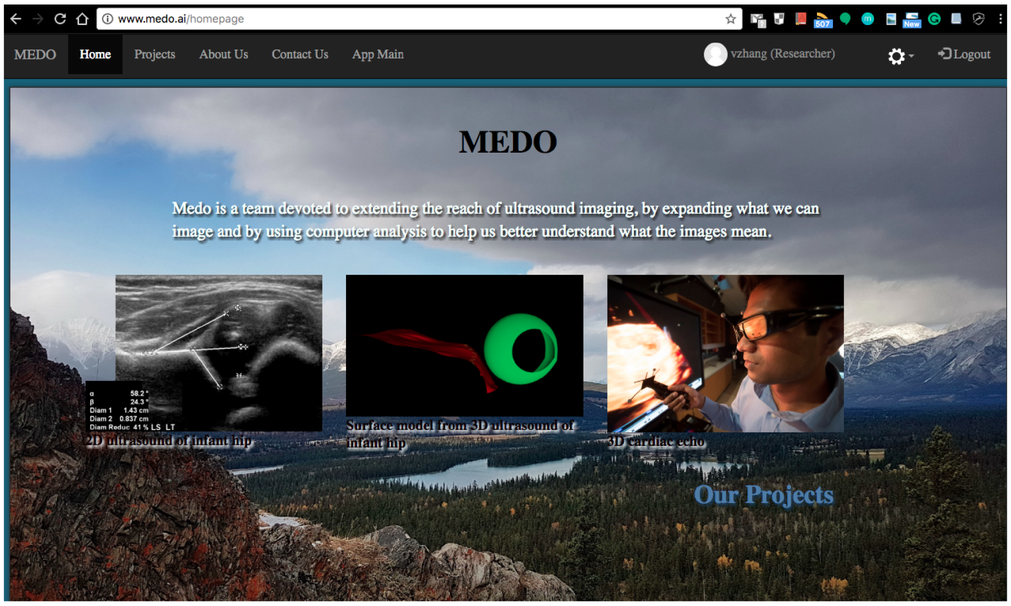

www.medo.ai/homepage
MEDO
Home
Projects
About Us
Contact Us
App Main
vzhang (Researcher)
Logout
MEDO
Medo is a team devoted to extending the reach of ultrasound imaging, by expanding what we can image and by using computer analysis to help us better understand what the images mean.
2D ultrasound of infant hip
Surface model from 3D ultrasound of infant hip
3D cardiac echo
Our Projects

Figure 4.7: The home page of the web application

4.5 Real-world Deployment: A Web-based Medical Image Analysis System

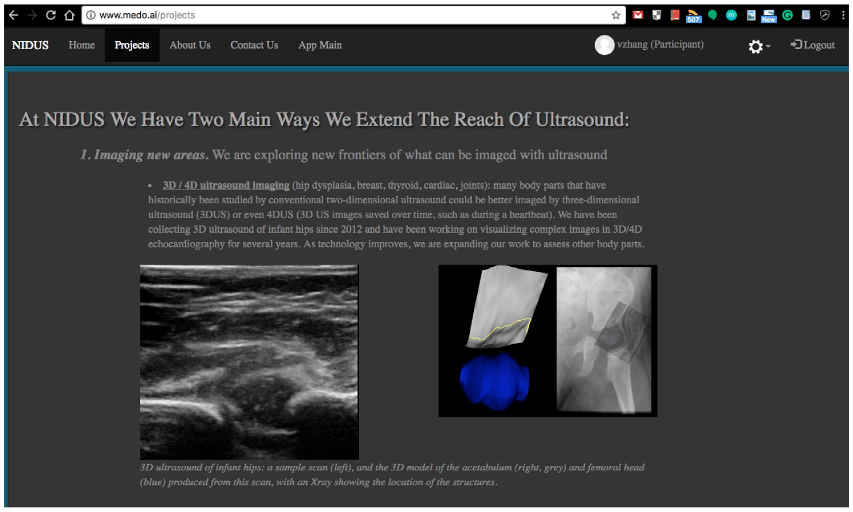
We have deployed the proposed algorithm in a web application for cloud-based acetabulum segmentation. The web application can be accessed at `http://www.medo.ai`, shown in Fig.4.7.

As described in the experiment section, ultrasound scan is an effective way to identify possible DDH of the hip. It is necessary to perform this test early, preferably for infants. The consequences of missing the condition are costly and debilitating: if diagnosis of DDH is missed until even one year of age, a series of increasingly complex surgeries can be indicated, with poorer success rates the later the diagnosis is made. If untreated, even mild DDH can lead to a life of suffering from premature hip osteoarthritis. The ultimate goal of this project is to make the collection and analysis of the ultrasound scan widely available to the public. We aim to create a set of tools where anyone can collect and analyze the data from home rather than going in a hospital to perform special test. For the analysis, the scan would be uploaded to a cloud server where deep-learning algorithm instantly returns the most appropriate diagnosis. Every case uploaded to the cloud is itself used to further train the deep-learning algorithm, constantly refining accuracy. Within a generation the societal burden of hip osteoarthritis could drop substantially, with large cost savings.

As one algorithm in this on-line tool set, the proposed segmentation algorithm has been trained off-line and deployed on the cloud server. It can auto-segment acetabulum from the images uploaded by the users and visualize the segmentation result. Fig. 4.9 demonstrates a few examples of the segmentation results for test images, where the regions in red illustrate the predicted acetabulum. The user interface is shown in Fig. 4.8.

The web application is in its early shape. Two modes have been implemented so far: single-image mode and volume mode. In the single-image mode, the user picks the 2D slices that he or she would like to segment, and get back the 2D segmentations for these slices. In the volume mode, the goal is to quickly obtain the segmentation for the entire 3D volume. The algorithm first picks several "key-slices" automatically from the volume and generates their segmentations (for example, segment one out of every 20 slices). Then the segmentations for the rest of the slices in between the "key-slices" are obtained by interpolation.

The algorithm currently runs at about 0.35 seconds per 2D image on a Tesla M60 GPU on an AWS server. And the average time taken in the volume mode is roughly 5 seconds per volume. We are working on further integrating the algorithm and improving its speed and segmentation accuracy.

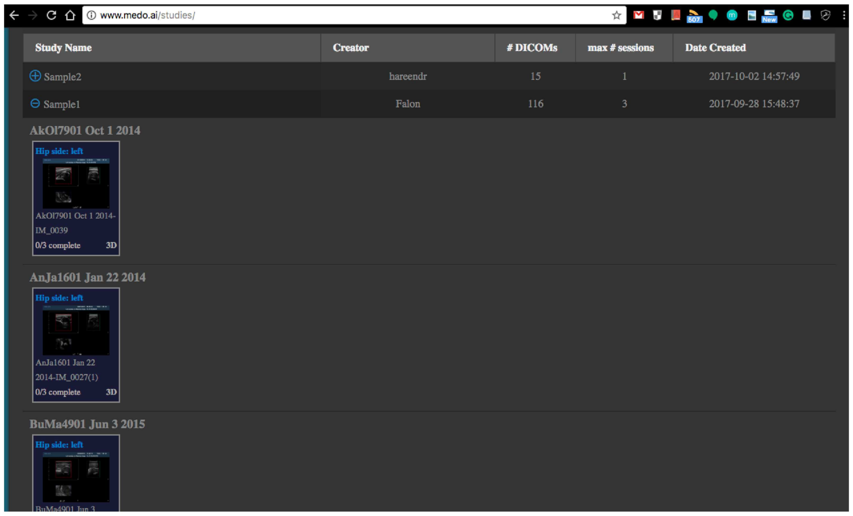


At NIDUS We Have Two Main Ways We Extend The Reach Of Ultrasound:

1. Imaging new areas. We are exploring new frontiers of what can be imaged with ultrasound

- **3D / 4D ultrasound imaging** (hip dysplasia, breast, thyroid, cardiac, joints): many body parts that have historically been studied by conventional two-dimensional ultrasound could be better imaged by three-dimensional ultrasound (3DUS) or even 4DUS (3D US images saved over time, such as during a heartbeat). We have been collecting 3D ultrasound of infant hips since 2012 and have been working on visualizing complex images in 3D/4D echocardiography for several years. As technology improves, we are expanding our work to assess other body parts.

3D ultrasound of infant hips: a sample scan (left), and the 3D model of the acetabulum (right, grey) and femoral head (blue) produced from this scan, with an Xray showing the location of the structures.



Study Name	Creator	# DICOMs	max # sessions	Date Created
⊕ Sample2	hareendr	15	1	2017-10-02 14:57:49
⊖ Sample1	Falon	116	3	2017-09-28 15:48:37

AkOl7901 Oct 1 2014

Hip side: left
AkOl7901 Oct 1 2014-IM_0039
0/3 complete 3D

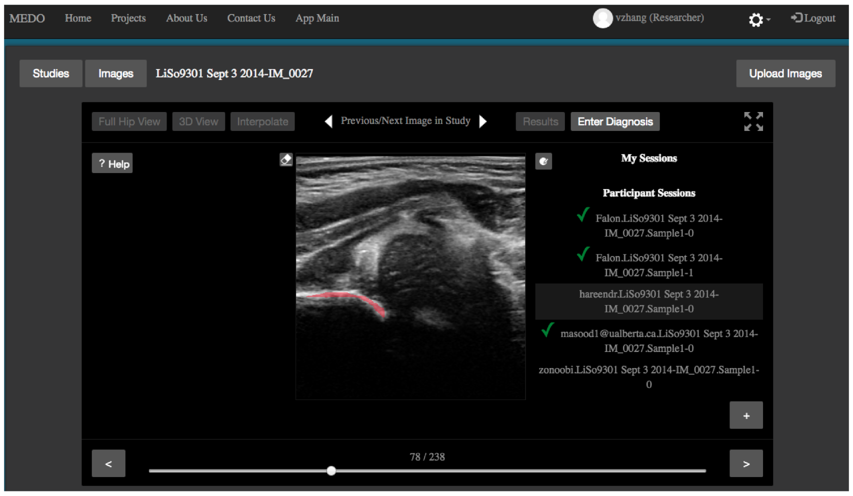
AnJa1601 Jan 22 2014

Hip side: left
AnJa1601 Jan 22 2014-IM_0027(1)
0/3 complete 3D

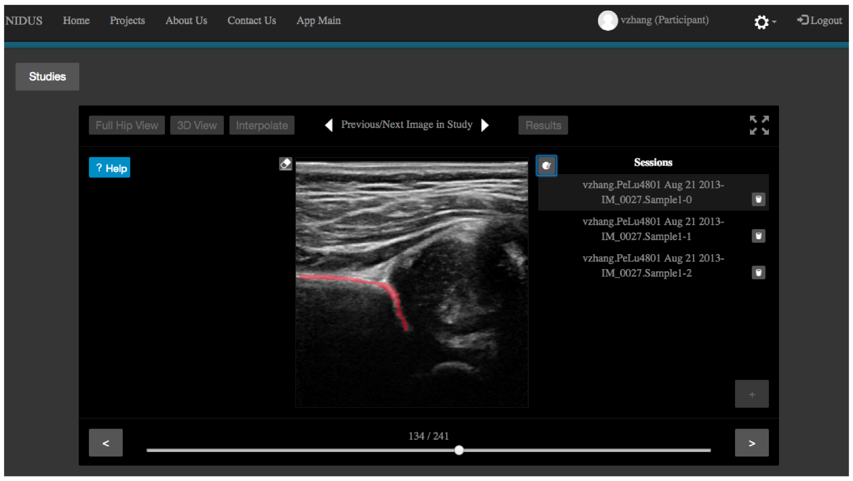
BuMa4901 Jun 3 2015

Hip side: left

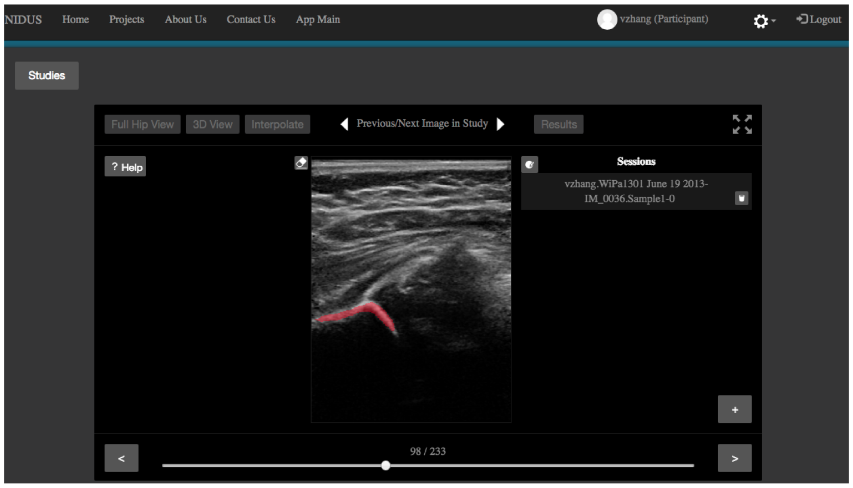
Figure 4.8: Web interface


MEDO
Home
Projects
About Us
Contact Us
App Main
vzhang (Researcher)
Logout
Studies
Images
LiSo9301 Sept 3 2014-IM_0027
Upload Images
Full Hip View
3D View
Interpolate
Previous/Next Image in Study
Results
Enter Diagnosis
? Help
My Sessions
Participant Sessions
Falon.LiSo9301 Sept 3 2014-IM_0027.Sample1-0
Falon.LiSo9301 Sept 3 2014-IM_0027.Sample1-1
hareendr.LiSo9301 Sept 3 2014-IM_0027.Sample1-0
masood1@ualberta.ca.LiSo9301 Sept 3 2014-IM_0027.Sample1-0
zonoobi.LiSo9301 Sept 3 2014-IM_0027.Sample1-0
+
<
78 / 238
>

(a) Test Image 1


NIDUS
Home
Projects
About Us
Contact Us
App Main
vzhang (Participant)
Logout
Studies
Full Hip View
3D View
Interpolate
Previous/Next Image in Study
Results
? Help
Sessions
vzhang.PeLu4801 Aug 21 2013-IM_0027.Sample1-0
vzhang.PeLu4801 Aug 21 2013-IM_0027.Sample1-1
vzhang.PeLu4801 Aug 21 2013-IM_0027.Sample1-2
+
<
134 / 241
>

(b) Test Image 2



(c) Test Image 3

Figure 4.9: Segmentation examples on the webapp

Chapter 5

Conclusion and Future Work

We presented a novel method for improving the segmentation accuracy of fully convolutional networks. Drawing inspiration from the object detection literature, we incorporated the Region-Proposal-Network (RPN) which was part of the backbone of Faster R-CNN. The bounding box prediction from RPN was used as an extra input to the downstream convolutional layers for segmentation to improve its localization capability.

We introduced a ROI convolutional layer that directly convolved on all the ROIs in one pass without the need of ROI pooling or region-wise processing, and showed that this layer was trainable through backpropagation.

We essentially broke down the segmentation task into two steps: first localizing a rough region in which the object had been detected, then performing segmentation in the region. The proposed network was trained end-to-end and tested on two image segmentation datasets. We showed that by breaking down the segmentation problem into a joint detection and segmentation process, the segmentation accuracy was improved on small objects.

For future work, it would be interesting to extend this work from binary segmentation to multi-class segmentation. It may also be worth investigating adding the gradients w.r.t. the bounding box parameters in the back-propagation of the ROI convolution.

References

[1] *Ama geoffrey hinton*, `https://www.reddit.com/r/MachineLearning/comments/2lmo0l/ama_geoffrey_hinton/`, Accessed: 2017-11-05. 13

[2] P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik, "Semantic segmentation using regions and parts," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, IEEE, 2012, pp. 3378–3385. 17

[3] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 328–335. 17

[4] S. Bauer, L.-P. Nolte, and M. Reyes, "Fully automatic segmentation of brain tumor images using support vector machine classification in combination with hierarchical conditional random field regularization," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2011, pp. 354–361. 9

[5] T. Brosch, Y. Yoo, L. Y. Tang, D. K. Li, A. Traboulsee, and R. Tam, "Deep convolutional encoder networks for multiple sclerosis lesion segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2015, pp. 3–11. 10

[6] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Transactions on image processing*, vol. 10, no. 2, pp. 266–277, 2001. 9

[7] D. Chen, G. Hua, F. Wen, and J. Sun, "Supervised transformer network for efficient face detection," in *European Conference on Computer Vision*, Springer, 2016, pp. 122–138. 13, 23

[8] J. J. Chen, N. J. Menezes, and A. D. Bradley, "Opportunities for crowdsourcing research on amazon mechanical turk," *Interfaces*, vol. 5, no. 3, 2011. 1

[9] 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*, vol. PP, no. 99, pp. 1–1, 2017, ISSN: 0162-8828. DOI: `10.1109/TPAMI.2017.2699184`. 7, 8

[10] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2016, pp. 424–432. 10, 35

[11] D. Cobzas, N. Birkbeck, M. Schmidt, M. Jagersand, and A. Murtha, "3d variational brain tumor segmentation using a high dimensional feature set," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, IEEE, 2007, pp. 1–8. 9

[12] J. Dai, K. He, and J. Sun, "Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1635–1643. 17, 19

[13] 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*, 2015, pp. 3992–4000. 17, 23

[14] 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*, 2016, pp. 3150–3158. 14

[15] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *ArXiv preprint arXiv:1603.07285*, 2016. 27, 28

[16] N. Dvornik, K. Shmelkov, J. Mairal, and C. Schmid, "Blitznet: A real-time deep network for scene understanding," *CoRR*, vol. abs/1708.02813, 2017. [Online]. Available: `http://arxiv.org/abs/1708.02813`. 16

[17] A. Foncubierta Rodríguez and H. Müller, "Ground truth generation in medical imaging: A crowdsourcing-based iterative approach," in *Proceedings of the ACM multimedia 2012 workshop on Crowdsourcing for multimedia*, ACM, 2012, pp. 9–14. 2

[18] E. Geremia, O. Clatz, B. H. Menze, E. Konukoglu, A. Criminisi, and N. Ayache, "Spatial decision forests for ms lesion segmentation in multi-channel magnetic resonance images," *NeuroImage*, vol. 57, no. 2, pp. 378–390, 2011. 9

[19] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448. 12, 22

[20] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: Data mining, inference, and prediction, second edition*, ser. Springer Series in Statistics. Springer New York, 2009, ISBN: 9780387848587. 23

[21] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin, and H. Larochelle, "Brain tumor segmentation with deep neural networks," *Medical image analysis*, vol. 35, pp. 18–31, 2017. 9, 10

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” *ArXiv preprint arXiv:1703.06870*, 2017. 12, 15, 23, 28

[23] 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*, 2016, pp. 770–778. 4, 7, 20

[24] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, “Densely connected convolutional networks,” *ArXiv preprint arXiv:1608.06993*, 2016. 7

[25] J. L. Jaremko, M. Mabee, V. G. Swami, L. Jamieson, K. Chow, and R. B. Thompson, “Potential for change in us diagnosis of hip dysplasia solely caused by changes in probe orientation: Patterns of alpha-angle variation revealed by using three-dimensional us,” *Radiology*, vol. 273, no. 3, pp. 870–878, 2014. 31

[26] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, “The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation,” in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, IEEE, 2017, pp. 1175–1183. 7, 8

[27] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in *Proceedings of the 22nd ACM international conference on Multimedia*, ACM, 2014, pp. 675–678. 35

[28] K. Kamnitsas, C. Ledig, V. F. Newcombe, J. P. Simpson, A. D. Kane, D. K. Menon, D. Rueckert, and B. Glocker, “Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation,” *Medical image analysis*, vol. 36, pp. 61–78, 2017. 9, 10

[29] M. Kampffmeyer, A.-B. Salberg, and R. Jenssen, “Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 1–9. 10, 36

[30] P. Krähenbühl and V. Koltun, “Geodesic object proposals,” in *European Conference on Computer Vision*, Springer, 2014, pp. 725–739. 17

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105. 4

[32] M. Lai, “Deep learning for medical image segmentation,” *ArXiv preprint arXiv:1505.02000*, 2015. 10

[33] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, “Fully convolutional instance-aware semantic segmentation,” *ArXiv preprint arXiv:1611.07709*, 2016. 14

[34] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017. 20

[35] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *ArXiv preprint arXiv:1708.02002*, 2017. 16

[36] 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 *European conference on computer vision*, Springer, 2014, pp. 740–755. 1

[37] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *ArXiv preprint arXiv:1702.05747*, 2017. 9

[38] 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*, Springer, 2016, pp. 21–37. 16

[39] 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*, 2015, pp. 3431–3440. 5, 6, 24, 34

[40] L. Maier-Hein, S. Mersmann, D. Kondermann, S. Bodenstedt, A. Sanchez, C. Stock, H. G. Kenngott, M. Eisenmann, and S. Speidel, "Can masses of non-experts train highly accurate image classifiers?" In *International conference on medical image computing and computer-assisted intervention*, Springer, 2014, pp. 438–445. 2

[41] Martín Abadi, Ashish Agarwal, Paul Barham, *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015. [Online]. Available: `https://www.tensorflow.org/`. 35

[42] M. Moradi, Y. Guo, Y. Gur, M. Negahdar, and T. Syeda-Mahmood, "A cross-modality neural network transform for semi-automatic medical image annotation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II*, S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, Eds. Cham: Springer International Publishing, 2016, pp. 300–307, ISBN: 978-3-319-46723-8. 2

[43] A. Prasoon, K. Petersen, C. Igel, F. Lauze, E. Dam, and M. Nielsen, "Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2013, pp. 246–253. 9, 10

[44] 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*, 2016, pp. 779–788. 16

[45] 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*, 2015, pp. 91–99. 11, 21–24, 28, 34

[46] *Robotic instrument segmentation challenge*, `https://endovissub2017-roboticinstrumentsegmentation.grand-challenge.org`, Accessed: 2017-12-17. 33

[47] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2015, pp. 234–241. 7, 8, 35

[48] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015. 1

[49] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ArXiv preprint arXiv:1409.1556*, 2014. 4, 7, 20

[50] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Computer Vision and Pattern Recognition (CVPR)*, 2015. 4

[51] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013. 17

[52] N. Xu, N. Ahuja, and R. Bansal, "Object segmentation using graph cuts based active contours," *Computer Vision and Image Understanding*, vol. 107, no. 3, pp. 210–224, 2007. 9

[53] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" In *Advances in neural information processing systems*, 2014, pp. 3320–3328. 5

[54] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *ArXiv preprint arXiv:1511.07122*, 2015. 7

[55] L. Zhang, L. Lin, X. Liang, and K. He, "Is faster r-cnn doing well for pedestrian detection?" In *European Conference on Computer Vision*, Springer, 2016, pp. 443–457. 13

[56] Y. Zhang, X. Chen, J. Li, C. Wang, and C. Xia, "Semantic object segmentation via detection in weakly labeled video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3641–3649. 18