Western Graduate&PostdoctoralStudies

Western University

Scholarship@Western

Electronic Thesis and Dissertation Repository

4-10-2017 12:00 AM

Improving Deep Learning Image Recognition Performance Using Region of Interest Localization Networks

AbdulWahab Kabani
The University of Western Ontario

Supervisor
Mahmoud R. El-Sakka
The University of Western Ontario

Graduate Program in Computer Science
A thesis submitted in partial fulfillment of the requirements for the degree in Doctor of Philosophy

© AbdulWahab Kabani 2017

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Artificial Intelligence and Robotics Commons

Recommended Citation

Kabani, AbdulWahab, "Improving Deep Learning Image Recognition Performance Using Region of Interest Localization Networks" (2017). *Electronic Thesis and Dissertation Repository*. 4485.
https://ir.lib.uwo.ca/etd/4485

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

Abstract

Deep Learning has been gaining momentum and achieving the state-of-the-art results on many visual recognition problems. The roots of this field can be traced back to the 1940s of the 20th century. The field has recently started delivering some interesting results on many image understanding problems. This is mainly due to the availability of powerful hardware that can accelerate the training process. In addition, the growth of the Internet and imaging devices such as mobile phones and cameras has contributed to the increase in the amount of data that can be used to train neural networks. All of these factors have contributed to the success of deep learning on large scale image understanding tasks.

Many image understanding problems do not have large training data. This is especially true in many special purpose datasets such as medical images, astronomical images, and environmental images. These application do not have large training datasets because unlike natural images, users do not typically take these images and upload them to the web. In addition, some of these applications, such as medical imaging, have many restrictions on sharing the data in order to protect the privacy of the patients. Finally, the labeling process needed for training natural images can be done by any person, unlike special purpose datasets. For example, in medical imaging, the images must be labeled by medical or clinical experts in the field. This results in datasets that are normally much smaller than natural images datasets as these experts have limited time to invest in the creation of the training sets.

Luckily, in many of these applications, the most discriminative features may be present in a small region of interest. In this work, we present a method of training deep learning models on problems with low number of training images. We will do that by localizing a region of interest in these images, which will help reduce the problem of overfitting. In this thesis, two localization architectures are introduced, namely: the standard localization network and the wide localization network (wide net). The latter has several advantages which we explain thoroughly.

The first problem we will introduce is the Right whale recognition problem. The problem involves recognizing whales from aerial images by analyzing the callosities pattern on their heads. We will study how localizing the region of interest can be used to make deep learning work on such a small dataset.

The second problem we will study is the estimation of the ejection fraction and left ventricle volume by analyzing cardiac MRI images. Automatically estimating the ejection fraction and volume of the heart can help in identifying and diagnosing several cardiac health issues. Similarly, this dataset contains only a small number of training subjects.

Keywords: Localization, Detection, Recognition, Artificial Neural Networks, Deep Learning, Convolutional Neural Network, Image Classification, Whale Localization, Whale Detection, Whale Recognition, Cardiac MRI, Left Ventricle, Automatic Ejection Fraction Estimation

Co-Authorship Statement

During my PhD studies, I was the main author in 8 papers. This thesis is a summary of 4 papers out of those 8 papers. Appendix A contains a list of the papers that were published during my PhD. A copyright release information about the papers included in this thesis is available in Appendix B . I am the primary author in all of the papers that were published during my doctoral studies. In all of these papers, Dr. Mahmoud El-Sakka was the co-author.

Acknowlegements

I would like to thank Dr. Mahmoud El-Sakka for giving me the chance to do my graduate work under his supervision. His support made this work possible.

My family (father, mother, brother) were the source of my strength during the most difficult times during my studies. We have not seen each other for the past 6 years, I hope we meet again soon.

Contents

Abstract i

Co-Authorship Statement iii

Acknowlegements iv

List of Figures viii

List of Tables x

List of Appendices xi

List of Abbreviations, Symbols, and Nomenclature xii

1 Introduction 1

1.1 The Deep Learning Journey 1

1.2 Region of Interest Localization 7

1.3 Right whale Recognition 12

1.4 Left Ventricle and Ejection Fraction Estimation 15

2 Right whale Recognition using the Standard Localization Network 17

2.1 Introduction 17

2.2 Method Overview 18

2.3 Body Localization 19

2.4 Head Localization 24

2.5 Recognition 26
2.6 Implementation and Results 28
2.7 CONCLUSION 34

3 Right whale Recognition using the Wide Localization Network 35

3.1 Introduction 35
3.2 Method Overview 35
3.3 Head Localization 37
3.4 Bonnet and Blow Hole Localization 39
3.5 Head Orientation and Region of Interest Alignment 40
3.6 Recognition 41
3.7 Results 41
3.8 Conclusion 44

4 Heart Volume Fraction Prediction Using Standard Net 46

4.1 Introduction 46
4.2 Left Ventricle Localization 47
4.3 Preprocessing 49
4.4 Volume Estimation 51
4.5 Results 53
4.6 Conclusion 55

5 Heart Volume and Ejection Fraction Prediction Using Wide Net 56

5.1 Introduction 56
5.2 Localization 58
5.3 Determining Interest Images and Segmentation 61
5.4 Volume Estimation 62
5.5 Results 63
5.6 Conclusion 65

6 Conclusion **67**

Bibliography **70**

A List of Papers Published During My Doctoral Studies **82**

A.1 Publications included in this thesis: 82

A.2 Publications **not** included in this thesis: 83

B Copyright Release **84**

Curriculum Vitae **86**

List of Figures

1.1 Standard Net Architecture 9
1.2 The architecture of wide net 11
1.3 Upsampling By repetition 12

2.1 The overview of the method 19
2.2 A sample of images 20
2.3 Head Crops Comparison 21
2.4 Body Localization Overview 22
2.5 Localization architecture 23
2.6 Head Localization Overview 25
2.7 Recognition Architecture 26
2.8 ImagesNumberDistribution 29
2.9 Loss Curve 31
2.10 Body Localization Sample 32
2.11 Head Localization Sample 32
2.12 Bonnet Progress 33
2.13 Body Failed Cases 33
2.14 Head Failed Cases 34

3.1 The overview of the whale recognition method using wide net 36
3.2 The architecture of wide net 38
3.3 Region of Interest Alignment and Extraction 40
3.4 Recognition Architectures 41

3.5 A sample of Images 44

4.1 Model Overview 48
4.2 Heart MRI Localization architecture 49
4.3 Left Ventricle Localization Sample 50
4.4 Slice Localizer Network 51
4.5 Ordering the Slices 52
4.6 Volume Estimation Architecture 53
4.7 Training and Validation Loss 54

5.1 Model Overview 58
5.2 Network Architecture 59
5.3 Localization Sample 60
5.4 Segmentation Sample 61
5.5 Segmentation Loss 63

List of Tables

2.1 Data augmentation 27
2.2 Teams Ranking 30

3.1 Recognition Data augmentation 42
3.2 Results of all Models 43
3.3 Teams Ranking 45

4.1 Errors Summary 55

5.1 Diastole RMSE Comparison 65
5.2 Systole RMSE Comparison 65
5.3 Ejection Fraction RMSE Comparison 66

List of Appendices

Appendix A: List of Papers Published During My Doctoral Studies 82
Appendix B: Copyright Release 84

List of Abbreviations, Symbols, and Nomenclature

Shortcut	**Description**
CNN	Convolutional Neural Network
Convnet	Convolutional Neural Network
CPU	Central processing unit
D	Diastolic Volume
EF	Ejection Fraction
FCN	Fully Convolutional Neural Network
GPU	Graphics processing unit
IoU	Intersection over Union
LTSM	Long Short Term Memory Networks
MAE	Mean Absolute Error
Standard Net	The Standard localization neural network
RAM	Random Access Memory
ReLU	Rectified Linear Units
ResNet	Residual Network
RMSE	Root Mean Squared Error
ROI	Region of Interest
S	Systolic Volume
VGG	Visual Geometry Group Neural Network
Wide Net	The Wide Localization Neural Network

Chapter 1

Introduction

1.1 The Deep Learning Journey

Researchers have always wanted to create machines that can perform intuitive tasks such as: understanding images, speaking, and interacting with humans. Ironically, even though these tasks are performed by humans with no effort, explaining how these tasks can be performed is not straightforward. Because of this, it is difficult for computers to perform these tasks. Indeed, computers are good at things that can be easily described by writing a piece of code. Generally speaking, there have been two approaches to creating machines that can perform *intelligent* tasks, namely: the *rule based* approach and the *experience based* approach. The *rule based* approach involves having humans specify a set of rules that describe the solution to the problem. On the other hand, the *experience based* approaches involves feeding data into the machine and having the computer learn from the experience over time. Over the past few years, the latter approach has been showing some positive results. *Deep Learning* [24] is a field in machine learning which involves stacking a set of layers together in order to learn to accomplish a task. The deep learning model can learn complex knowledge because of its hierarchical nature.

In the image classification problem, classical approaches rely on extracting features from the data (the images). These features are then passed to the model so that the model can learn to classify the image. This greatly reduces the number of parameters that need to be learned. In addition, the input to the model is a feature vector that contains a set of high level features that the human views as important. The success or failure of the learning process relies heavily on how good the features are. This has lead to the development of a lot of techniques and practices to perform *feature engineering* on the raw input data. *Feature engineering* and *information representation* are essential when dealing with many types of classifiers including logistic regression, SVM, etc. The Deep learning approach is however different. Normally, deep learning involves feeding raw data into the model and the model will learn low level representation of the data (in the bottom layers), medium level representation in the intermediate layers, and high level abstract representation at the top layers.

Deep learning is more suitable for problems where it is difficult to know what kind of features that should be extracted from the data. *Representation learning* [24] involves the process of learning not only the output of the problem but also the appropriate set of representations for the problem (the set of features that should be used to describe the problem). Indeed, these features may be really good that they can be used to solve other similar problems. For instance, a model trained to perform image classification can also be used to perform semantic segmentation [56]. In the visual recognition problem, deep learning performs really well because it is very difficult to extract reliable features from images. Factors such as image brightness and background occlusion can change the way objects appear in different images.

Deep learning is not a new field. In fact, deep learning dates back to the 1940s [59]. Perceptrons [70, 71] were developed in the 1950s and represent an early version of deep learning. It is important to note that while neuroscience was a main source of inspiration for the development of deep learning, none of the deep learning models are viable biological models. For instance, in Neuroscience, it was shown that neurons in the auditory processing region can be trained to "see" by rewiring the visual signals [84]. Indeed, there is a wide believe in neuroscience that one learning algorithm is sufficient to learn different kinds of tasks. In deep learning, this has not be shown to be true as of the time of writing this thesis. Some architectures are more suitable for certain tasks than others. For example, *convolutional neural networks* [51] are a special type of neural networks that are suitable for visual recognition problems. These networks involve having layers with limited connectivity to simulate a behavior similar to convolution. These types of networks date back to the 1980s [19]. On the other hand, *long short term memory networks* (LTSM) [34] are more suitable for natural language processing tasks.

As mentioned earlier, deep learning is not a new field. However, it only started delivering good results recently. There are two main reasons for the resurrection of the field. First, the introduction of powerful hardware that can be used to speed up the training process of the deep learning models. *Graphical Processing Units* (GPUs) are mainly used to accelerate the output to a computer display. However, over the past few years, they are being used in image processing, computer graphics, and deep learning. This is largely due to their parallel structure which is more suitable than the CPU when working on these types of applications. With respect to deep learning, these devices can train models with around 80x speed up compared to the general purpose CPU.

Second, the availability of large amounts of labeled training data made training deep learning models possible. This data (images, videos, sounds, etc...) are largely generated by users and hosted on the web. In other words, the availability of cameras, mobile phones, and computers contributed to the increase in the amount of training data available to train deep learning models. It is important to note that typically deep learning models require a lot of training data in order to achieve the desired results. Not providing enough training data can lead to overfitting. Overfitting happens when the model performs really well on the training data but fails to generalize well to unseen data. The introduction of large training data sets helped in boosting the performance of deep learning models. Some of these data sets include PASCAL VOC [16, 15], ADE20K [92] and the Imagenet challenge [11, 73].

Backpropagation [72, 46] is the main algorithm used for training deep neural networks. Backpropagation involves two main steps, namely: forward pass and weight update. In the first step, an input is passed through the network until it hits the output layer. Then an error value based on a predefined loss function is calculated and used to calculate the error of each neuron in the network. Finally, the error values associated with each neuron are used to compute the gradient of the loss function with respect to the current network weights. In the second step, the weights are updated based on the gradient that was computed.

Deep learning has been very successful over the past few years in supervised learning. The main difference between supervised and unsupervised training is that in the supervised setting, the desired output value is provided along with the training data. For instance, if we want to train a deep learning model to detect whether an image shows a malignant or benign tumour, we need to provide some images as training data. In a supervised setting, we also have to provide labels to describe the type of tumours in each of the training images. Most of the success that deep learning showed over the past few years was based on supervised learning. However, unsupervised learning has been a very active area of research [50, 66, 25, 27, 63].

As mentioned earlier, deep learning is achieving state-of-the-art results on many visual recognition problems [31, 79, 48, 54, 75]. In general, deep learning models require a lot of labeled data. Thanks to the availability of many datasets [16, 15, 92, 73], we can train deep learning models much easily. However, there is a common theme among these datasets, which is that they involve general natural images where any human can label the images in that dataset.

Unfortunately, this is not true for many types of problems. For instance, if we are interested in developing a model that can determine the type of cancer from medical images, the images in this dataset can only be labeled by a doctor or an expert in the field. Unlike natural images datasets, the dataset can only be labeled by people who are expert in the field. This makes creating these types of datasets very expensive. Furthermore, the size of many general images datasets is mainly due to having users uploading photos and sharing them on the web. This is not the case in many medical (and other special-purpose) datasets where creating these datasets can be complicated due to privacy concerns. That's why datasets that include medical images contain very few images (generally, in the order of 100s of images) as opposed to general images datasets such as Imagenet [11, 73] that includes millions of images.

There are many datasets with very small number of images. The galaxy zoo challenge [12, 85, 1] involves training models to classify the morphology of the galaxies. These images were acquired using a telescope and include around 60,000 images. The Diabetic Retinopathy [42] contains around 35,126 training images (around 17,563 per subject). The Right whale recognition dataset [43] that we will study contains only 4,544 training images. Finally, the left ventricle volume estimation dataset [41] that we will study has only training data for 500 subjects.

One way to bring the power of deep learning to smaller datasets is to perform transfer learning [32, 87, 64, 89]. However, there are several issues with this. First, pre-training a deep learning model on a large dataset such as the Imagenet [11, 73] may not be very effective on some special types of problems such as medical imaging. This is because the medical images are very different from the natural images. The second problem is a legal issue. Pre-training a model on Imagenet [11, 73] may be problematic because this dataset can only be used for research purpose.

CNNs have achieved the state-of-the-art results on many classification problems [48, 79, 31, 54, 75]. For instance, ResNet[31] is an architecture that can be trained with very large number of layers while being resilient to the vanishing gradient problem [3]. There has been an interest in translating this success into other tasks such as *detection* [67, 22, 68, 21] and *semantic segmentation* [56, 62, 17, 88, 29, 7, 90].

Most of the deep learning semantic segmentation methods are based on classification models [31, 79, 48, 54, 75] that were pre-trained on the Imagenet dataset [11, 73]. The weights of the pre-trained models are re-purposed in order to perform semantic segmentation. Semantic segmentation can be seen as a problem of dense prediction where each pixel in the image should be classified to one of the possible classes. Fully convolutional neural networks (FCN) [56] are deep learning models that can perform semantic segmentation. They are based on re-purposing the VGG model [75] which was pretrained to perform classification. Most of the models for semantic segmentation [56, 62, 17, 88, 29, 7, 90] rely on pre-trained classification models because the amount of training data available in classification problems far exceeds the amount of training data for semantic segmentation even though the latter is more challenging. This is because labeling images for segmentation tasks is very time consuming. The simplest way to make a classification model work for semantic segmentation involves dropping the dense layers and replacing them with 2D convolutional layers.

Most of the current semantic segmentation models are based on Fully Convolutional Neural Networks (FCN) [56]. In [86], a detailed study about the use of ResNet [31] in semantic segmentation was conducted. An attention mechanism that can learn to weight the multi-scale features at each pixel location was proposed in [8]. Typically, in these types of models, they accept a multi-scale input (an image resized to different scales). They perform really well on objects at small scales. PSPNet [90] showed that using global and local clues about the image can boost the performance on the image segmentation challenge.

It may be useful to increase the size of the receptive field. *Dilated convolution* [88, 7] can be used to achieve this. The inspiration came from a signal processing algorithm for wavelet decomposition called algorithme a trous [35]. In [88], it was shown that *Dilated convolution* has nice properties that makes it ideal for dense prediction in semantic segmentation. The main advantage of using dilated convolution is that it can be used to aggregate multi-scale information while maintaining a low number of parameters. The architecture is composed of 8 layers with different dilation rates. This architecture can be embedded into another base architecture that was trained on the imagenet challenge [11, 73]. Some semantic segmentation models [28] rely on the intermediate layers because they tend to carry a relatively high resolution information in addition to the abstract information that can be used to perform the classification. RefineNet [53] was proposed as a multi-path network in order to exploit features at different levels. Finally, the semantic segmentation performance of any model can be improved using conditional random fields (post-processing operation) as shown [91].

There are various ways to deal with the fact that layers at the top have low resolution. In

[62, 13, 7], deconvolution was used to upsample the feature maps from the top layer in order to increase their resolution. The other way is to use dilated convolution [88] which is similar to convolution with holes in between. This allows for capturing multi-scale features without downsampling and in a relatively efficient way.

1.2 Region of Interest Localization

While the methods mentioned previously achieve excellent results on semantic segmentation, all of them rely on pre-trained data to achieve the required results. As a consequence, they have a large GPU RAM requirement because they depend on adding layers to models that were trained on classification tasks. We are interested in developing region of interest (ROI) model that have lower memory requirement. These models will be used as a means of pre-processing the data to simplify the classification problem (in this thesis: the whale recognition problem (Chapters 2 and 3) and the left ventricle volume estimation problem (Chapters 4 and 5)). Therefore, these models need to deliver the results as fast as possible and with low memory consumption. In other words, detecting the region of interest is a sub-task that can simplify the main problem (Right whale recognition or left ventricle estimation) and we are interested in an efficient solution that can perform this task.

This work will introduce two deep learning architectures that can be used for region of interest (ROI) localization. We will see how these networks can be used to improve the performance on two problems: Right whale recognition and left ventricle volume estimation. These architectures were developed on a laptop GPU with around 3.5 GB of available RAM. It is worth noting that laptop GPUs are around 5-6 time slower than desktop GPUs. While developing these two architectures, fast convergence and low memory requirement were the two main restrictions that influenced the design of the networks. In addition, because these networks will be used as a means for pre-processing the data, it is essential that they accomplish the task with relatively low computation power. In addition, one of the most important objectives that we wanted to achieve is being able to perform region of interest localization with different shapes whether the region of interest is a point, circle, triangle, rectangle, or irregularly shaped region of interest.

We introduce two architectures that can be used for localizing regions of interest (ROI) in an image. The two architectures are: the standard network (standard net) and the wide localization network (wide net). Some of the factors that influenced the design of these networks are the following:

- Because the networks will be used as a pre-processing step, they should be trained in a relatively short period of time.
- The networks should work well on machines with relatively low computing power.
- The networks should be able to learn an ROI of arbitrary shape (not just rectangle ROI).

Figure 1.1 shows the architecture of the standard net applied on two separate problems that we will cover in this work. In the first row, the network was trained to localize the bonnet (a point located on the head of the whale). The second row shows how the network can be used to predict the location of the left ventricle. As shown in the two problems, the network can have different input sizes depending on the problem.

Let's take the first row in Figure 1.1 as an example. The input of the architecture is an image of size 128×192. The ground truth is a flattened mask of size $128 \times 192 = 24,576$ pixels. In other words, the output of the network is a layer with 24,576 possible classes. The output layer is simply a flattened mask and reshaping this layer gives us back the predicted mask. The pixels with the highest intensities represent the location of the interest point.

The loss function we minimize is the categorical cross-entropy (or mutli-class logloss). The activation for each convolutional layer is ReLU [61] while the activation for the output layer is softmax. Softmax activation ensures that each pixel in the predicted mask is in the range of (0,1) and the pixels sum up to 1. In other words, it is a measure of certainty that a certain pixel in the predicted mask corresponds to the location of the interest point.

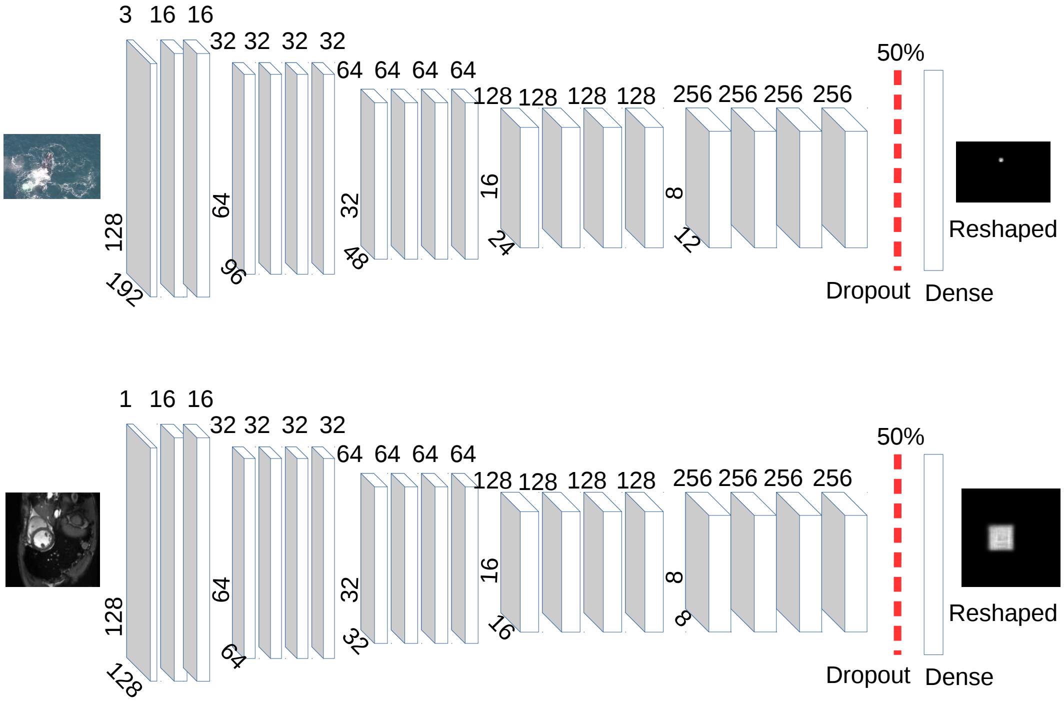

Figure 1.1: The standard net architecture: this figure shows the architecture of the standard localization net. In the first row, a input image is passed to the network and the bonnet (a point located on the head of the whale) can be localized after training the network. In the second row, the network predicts the location of the left ventricle from a cardiac MRI image. The network can have different input sizes depending on the problem.

Because the last layer in the network is softmax, pixels in the predicted mask are probabilities that range between 0 and 1 and that sum up to 1. It is very important to ensure that the ground truth (the labels) follow the same rules. In order to do that, we rescale the true mask (or the true labels) by dividing each pixel by the sum of the mask as shown in the Equation 1.1:

$$y_{ij} = \frac{pixel_{ij}}{\sum_{i=1}^{H}\sum_{j=1}^{W} pixel_{ij}} \tag{1.1}$$

where $pixel_{ij}$ is the pixel value at row i and column j. y_{ij} is the normalized pixel value such that $y_{ij} \in [0, 1]$ and $\sum_{i=1}^{H}\sum_{j=1}^{W} y_{ij} = 1$.

It is worth mentioning that if the pixels in the true masks are not rescaled as shown in

Equation 1.1, the network will not converge.

There are several limitations to the standard localization network. Here are some of them:

- Because the last layer is dense (fully connected), the number of parameters can grow very quickly.
- The network cannot be trained to learn more than one ROI. For instance, when applying this network to the whale recognition problem. We had to train two networks: one is used to localize the bonnet and the other is used to localize the blow hole.
- The network cannot learn to segment an arbitrary shaped ROI. This is because while the top layers has highly abstract information, the spatial resolution is lost due to the pooling layers.
- The output mask requires thresholding in order to produce a binary mask. The results may vary depending on the thresholding model.

The wide localization network (wide net) is a better alternative to the standard localization network. The architecture of this model is shown in Figure 1.2. This network is capable of performing segmentation and producing region of interests at a very low processing power. The network is composed of three levels: Level_1 is simply a set of convolutional and pooling layers. Level_2 involves merging the last layer from each block in level_1 in three different ways. Three types of merging is done: type_1 involves merging (block_2, block_3, block_4, block_5), type_2 involves merging (block_3, block_4, block_5), and finally type_3 involves merging (block_4, and block_5). Type_1 carries a combination of high resolution features along with highly abstract features. On the other hand, Type_3 carries only abstract features. Finally, in level_3, the outputs of level_2 are combined and a mask is produced.

The last layer in the network has a sigmoid activation to map the pre-activation values into values between 0 and 1. Rectified linear units (ReLU) activation [61, 30] are used after each layer. Each pre-activation is followed by batch normalization [36]. Channel-wise Batch normalization [36] accelerates the training process significantly. In order to extract more abstract

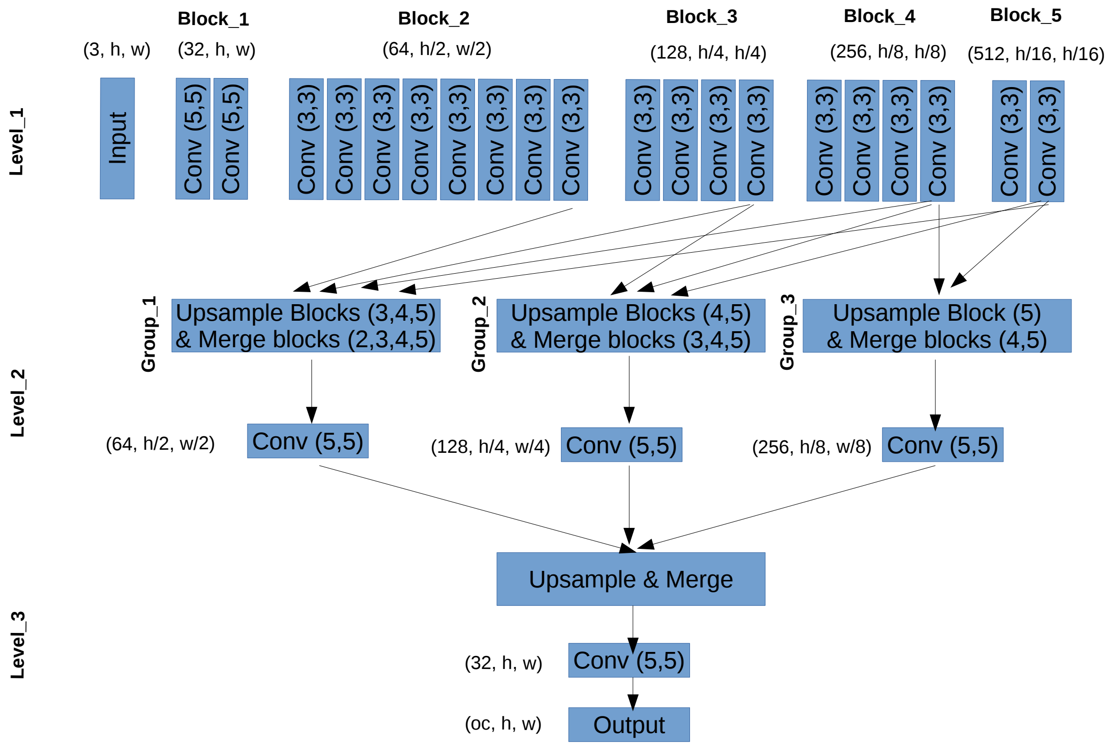

Figure 1.2: The architecture of wide net: the network is composed of three levels. Level_1 is just like any typical recognition network. Level_2 involves merging the last layer from each block in level_1 in three different ways. Finally, in level_3, the outputs of level_2 are combined and a mask is produced. Abbreviations: h is the height of the image, w is the width of the image, oc is the output channel size.

features, we use Max-pooling in level_1. As a result, the layers in each block in level_1 will have different sizes. In order to combine them together in level_2, we use upsampling to ensuring that the layers can be merged. This network has a low processing footprint and was previously tested on a device with only 3.5 GPU memory.

It is worth noting that in order to merge the layers, we need to upsample them. In order to reduce the GPU memory consumption, the upsampling process involves repeating the units across the height and width of the layer as shown in Figure 1.3. This greatly reduces the memory requirement because the upsampling process is not parametrized. In order to reproduce a high resolution region of interest, the upsampled layer is combined with layer(s) with higher

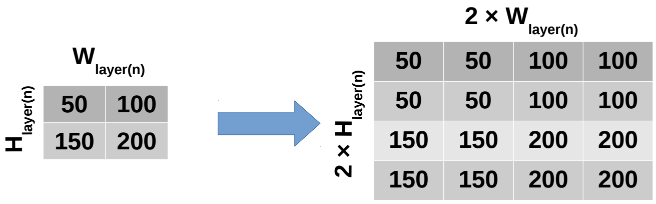

Figure 1.3: The units in a layer can be upsampled by repetition. The unit is repeated across two dimensions to upsample the layer. This is a non-parameterized operation that does not require parameters optimization during training. In order to reproduce a high resolution region of interest, the upsampled layer is combined with layer(s) with higher resolution.

resolution.

There are several advantages to this architecture over the standard architecture:

- It is a very efficient architecture with lower number of parameters.
- The network may have several output channels. This means that it can be trained to localize several ROIs simultaneously.
- The network can learn to identify an arbitrary shapped ROI including segmentation.
- The output mask can be thresholded simply by using the threshold value of 50%. This is because the last layer has a sigmoid activation.

1.3 Right whale Recognition

Right whales are endangered species with a population of around 450 whales. In Chapters 2 and 3, we will present two solutions based on deep learning which can be used to recognize Right whales from aerial survey images. The solutions were developed on a dataset provided by the National Oceanic and Atmospheric Administration [45, 43]. The main challenge we faced when working on this data set is that the size of the training set is very small (4,544 images) with some classes having only 1 image. While this dataset is by far the largest of its kind, it is very difficult to train a deep neural network with such a small data set. The highest accuracy we were able to achieve on this dataset when training on the images directly (without localization) is 8%. In order to improve the results and overcome this limitation, we can train models to localize the head of the whales. After that, we can train models to recognize the whales. This greatly reduces the amount of viewpoint variance making it possible to train a classification model to recognize the whale.

The main reason for why the dataset is small is because Right whales are endangered species. In addition, labeling these images require a marine biologist who is expert in these animals to perform this task. Unlike general and natural images, labeling these types of images is expensive because a random person cannot be used to label the data.

Historically, Right whales were the subject to harsh hunting since the 17th century. Some researchers believe that the name 'Right whale' comes from the fact that this is the 'right' type of whale for hunting. These whales were considered to be ideal for hunting for many reasons such as their tendency to live close to the shore, being rich in whale oil, and the fact that their bodies float when killed. Despite becoming protected species since 1949, the population is still endangered with being entangled in fishing gear, and collision with ships accounting for around 50% of deaths.

Right whales can be recognized by studying the callosities pattern on their heads. Manually classifying whales is a very time consuming process and automating this process can help scientists focus on their conservation efforts. Each Right whale has a unique callosities pattern on its head. These callosities pattern can be used to identify Right whales just like fingerprint pattern can be used to identify humans. We developed a model that can automatically recognize individual whales by analyzing the callosities pattern on their heads. The solution we describe in Chapter 2 was ranked 5th (out of 364 teams) in a Kaggle challenge [43] to solve this problem. This solution was based on the standard localization network. In Chapter 3, we improve the solution by using the wide net.

When working on this problem, we faced several challenges. First, the size of the training set is very small while the number of classes (individual whales) is large. Indeed, the number of individual whales (the classes we want to predict) is 447. Second, there is a huge variation in the clarity of each image. Finally, the size of each image is very large making it very difficult to load these images into the GPU.

Since the head of the whale occupies only a small area in the whole image, localizing the region of interest is very important. Localizing the region of interest can help the classification model focus on the most discriminative features and avoid irrelevant features such as features in the surrounding water. Normally, a deep learning classification model can learn to ignore irrelevant features by training it on large datasets. However, because this dataset [45] is very small with many classes including only one image, it is important to train the classification model only on the important features.

Here is a summary of the solutions that achieved the best results on this dataset. Deep learning have been very successful in recent years with visual recognition problems. However, it was very difficult to make it work for this type of data. This is mainly because the size of the data is very small. Anil Thomas from Nervana Systems [82] suggested locating the bonnet and the blow hole on the whale. The solution which is based on a deep learning library called Neon [78] used these two points to extract a patch that contains the whale's head. This approach proved to be very effective. First, it made the training process much faster because the training is being done on smaller images. Second, these head patches are better than the original images for training a deep learning model. This is mainly because training on these head patches made the model focus only on the most discriminative features (the head callosities) and ignore unimportant features such as features from the surrounding water.

To our knowledge, most of the solutions (including the two solutions described in Chapters 2 and 3) that ranked in the top on the competition's leaderboard followed this idea that was proposed in [82]. The team that was ranked in the second position [49] used a similar multi-stage approach. First, the head is localized by regressing a bounding box. Then, the head is aligned in an approach that is loosely inspired by a human face alignment approach that was introduced [81]. This team used a classifier that is an ensemble of deep neural networks with different variations of the VGG-Net [75] and ResNet [31].

The DeepSense.io team [5] that was ranked in the first position also used a multi-stage approach. This team produced a solution [5] that involves localizing the head of the whale and aligning it afterwards. They report that aligning the head of the whale is very important to achieve good results. Their final solution is based on combining the predictions of different deep learning models.

1.4 Left Ventricle and Ejection Fraction Estimation

The second problem we will discuss in chapters 4 are 5 is the problem of left ventricle volume and ejection fraction estimation. Cardiologists can assess cardiac function by analyzing the end-systolic and end-diastolic volumes, and ejection fraction. These values can be manually measured by a cardiologist but the process is slow and time consuming. We introduce an automated method that can estimate these values. We developed our method on a data set with 500 training studies, 200 validation studies, and tested it on 440 testing studies. The data set is compiled by the National Institutes of Health and Children's National Medical Center [41].

In general, left ventricle volume can be estimated by detecting it and segmenting its cavity [55, 57, 20, 44]. Once the cavity (the blood pool) is segmented, the volume can be calculated by summing up the sub-volumes of all slices according to simpson's rule.

In Chapter 4, we take a slightly different approach and estimate the volume using a convolutional neural network. In Chapter 4, we were able to estimate the volume with around +/- 15 ml mean absolute error. Furthermore, we were able to estimate the ejection fraction with +/- 5% mean absolute error. The ejection fraction is one of the most important cardiac measurements since it describes how good the heart is in pumping blood. The solution we present in Chapter 4 is based on the standard architecture. First, We localize the left ventricle. Then, the data is preprocessed, cleaned and standardized. Finally, the volume of the left ventricle is estimated.

In Chapter 5, we will present a solution based on the wide net architecture. We divide the task into several steps in order to ensure that the training process is successful. First, the left ventricle is localized. Then, the systole and diastole images for each slice are determined. After that, the left ventricle segmentation process is described. Finally, the volume of the left ventricle is estimated. After that, we will present the results. The approach we present in Chapter 5 achieves better results than the one achieved in Chapter 4 even though we only train the model on 25 studies.

Chapter 2

Right whale Recognition using the Standard Localization Network

2.1 Introduction

The North Atlantic Right whales [45] is an endangered species with around 450 whales left. Historically, Right whales were the subject to harsh hunting since the 17^{th} century. Some researchers believe that the name 'Right whale' comes from the fact that this is the 'right' type of whale for hunting. These whales were considered to be ideal for hunting for many reasons such as their tendency to live close to the shore, being rich in whale oil, and the fact that their bodies float when killed. Despite becoming protected species since 1949, the population is still endangered with being entangled in fishing gear, and collision with ships accounting for around 50% of deaths.

Right whales can be recognized by studying the callosities pattern on their heads. Manually classifying whales is a very time consuming process and automating this process can help scientists focus on their conservation efforts. We used a unique data set provided by the National

---

[0] The majority of this chapter was originally published in: AbdulWahab Kabani, and Mahmoud R. El-Sakka "North American Right whale Recognition using Very deep and Leaky Neural Network", Journal of Mathematics Applications, Vol 5., 2016. [38]

Oceanic and Atmospheric Administration [45],[43] to develop a model that can automatically recognize individual whales by analyzing the callosities pattern on their heads. The solution we describe in this paper was ranked 5^{th} (out of 364 teams) in a Kaggle challenge [43] to solve this problem.

When working on this problem, we faced several challenges. First, the size of the training set is very small while the number of classes (individual whales) is large. Second, there is a huge variation in the clarity of each image. Finally, the size of each image is very large making it very difficult to load these images into the GPU.

To overcome these challenges, we first localize and normalize the body with respect to rotation. After that, we localize the head of the whale. Finally, the whale is recognized using the callosities pattern on its head. These steps reduce the image size and ensure that we can fit the data into the GPU.

In this chapter, we will present general overview. Information about the data that we used will be presented in Section 2.2. Sections 2.3 and 2.4 describe the methods we used to localize the body and head of the whale, respectively. In Section 2.5, we present the model that we used to recognize the whales. The results are introduced in Section 2.6. Finally, we conclude our work in Section 2.7.

2.2 Method Overview

An overview of our method is presented in Figure 2.1. While the main task is to recognize the individual whales IDs, we could not do that directly on the original images. This is because the original images are huge and trying to fit them in the GPU during recognition is not feasible.

We divided the problem into smaller tasks (as shown in Figure 2.1). First, two models are trained to localize the bonnet and the blow hole, respectively. Then, using these points, the body of the whale is localized and rotated so that the body has angle=0 with the x axis and the head is pointing east. After that, a model is trained to localize the head of the whale. Finally, a

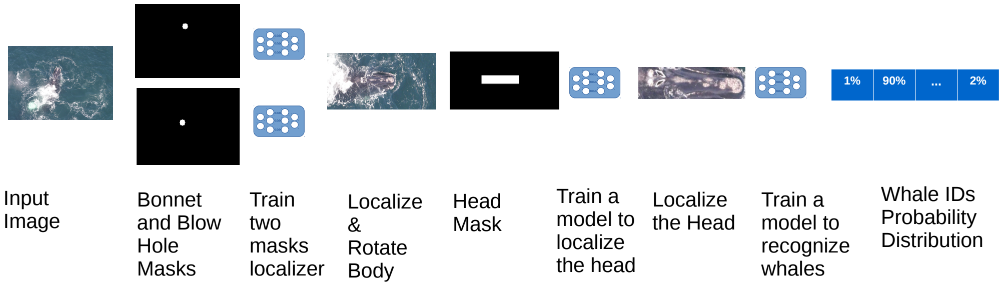
1%
90%
...
2%
Input Image
Bonnet and Blow Hole Masks
Train two masks localizer
Localize & Rotate Body
Head Mask
Train a model to localize the head
Localize the Head
Train a model to recognize whales
Whale IDs Probability Distribution

Figure 2.1: The overview of the method: First, two models are trained to localize the bonnet and the blow hole, respectively. Then, using these points, the body of the whale is localized and rotated so that the body has angle=0 with the x axis and the head is pointing east. After that, a model is trained to localize the head of the whale. Finally, a model is trained to produce a probability distribution over all possible whales.

model is trained to produce a probability distribution over all possible whales.

2.3 Body Localization

In this section, we describe how we trained a model to localize the body of the whale and normalize it with respect to rotation. The trained model will be able to take the original image as input and produce an output where the image is cropped and the whale is facing east. This is very important in order to train the classification model (presented in Section 2.5). Figure 2.2 shows a random sample of the input images that we pass to the model in order to localize the whale body.

This is a very important step for many reasons. First, localizing the body and the head (in Section 2.4) helps the model focus on the important features on the whale body rather than on features in the surrounding water. Because there are many classes with very few training images, focusing on the important features in the image (the callosities on the top of the head) is essential to alleviate overfitting.

Second, rotating the body such that the angle between the body and the x axis is 0 is important because it helps in extracting head crops with minimum amount of surrounding water. Otherwise, if the whale body has different orientation, extracting head crops may include

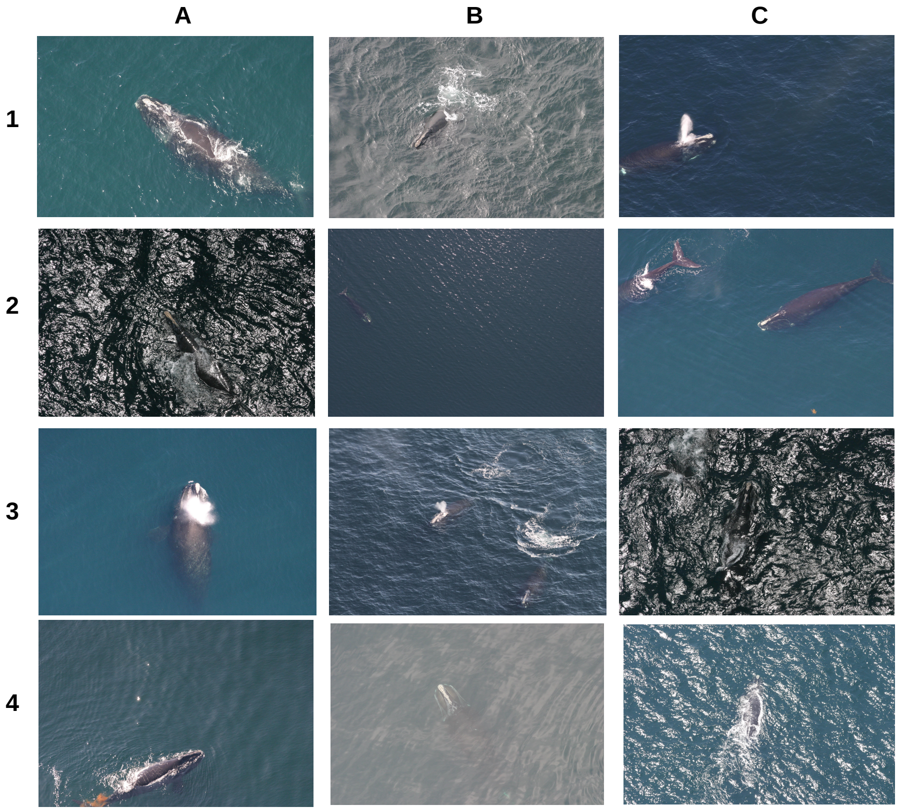

Figure 2.2: A random sample of images: These images were captured during several aerial surveys and under different lighting and environmental conditions (note images A1, B2, B4, C4). In addition to these variations, there are many others obstacles, which make these images very challenging. For instance, for many whales the head is not clear because the whale is blowing water as in images C1 and C3. Some images contain more than two whales like in images C2 and B3. In images A2 , C3, and B4, the foreground/background contrast is very low. Water reflection can make recognition very difficult like in image C4. The size of the whale with respect to the background (like the on in image B2) is another challenge.

a lot surrounding water. For example, Figure 2.3 shows two head crops, one crop is for a whale that has 0 degrees angle with the x axis and another one with around 135 degrees angle. It is clear that the one in the latter crop contains far more water.

To be a able to localize the body, we train two models: one will be used to recognize the bonnet (shown in Figure 2.3 as a red point) and the other is used to recognize the blow hole (shown in Figure 2.3 as a blue point). Using these two points, we can easily calculate the angle between the body and the x axis and rotate the body accordingly. This simple and powerful idea was originally introduced in [82] and we found it to be very helpful. In [82], two points are

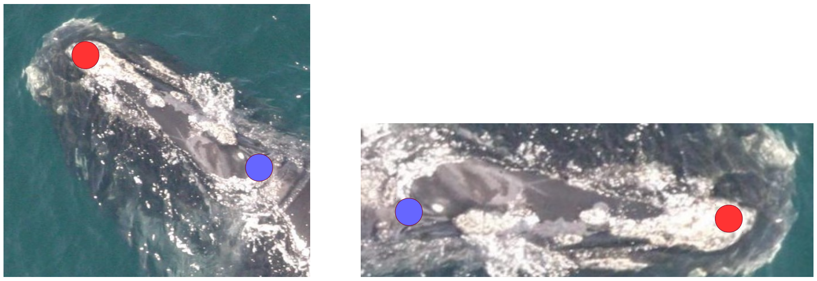

Figure 2.3: Comparison between two head crops. The one on the left is taken from an image where the angle of the body of the whale with the x axis is 135 degrees. On the other hand, the one on the right is the result of normalizing the angle of the body so that it is 0 degrees. The figure also shows the locations of the bonnet and the blow hole. Two localizations models are trained to recognize the bonnet and the blow hole, respectively. Then, using these two points, we can calculate the angle with the x axis and rotate the whale accordingly.

used to train two convolutional autoencoders. After that, the head was extracted and rotated. We use a similar approach. However, rather than training a convolutional autoencoder, we trained a deep neural network with the units in the output layer corresponding to individual pixels in the masked image. Figure 2.4 shows a summary of the body localization stage.

Each mask is used to train a deep neural network. The mask and the image are re-sized to size 128 (*height*)×192 (*width*).The labels (ground truth) for this network are the elements-wise multiplication of the resized mask with the gray scale image.

To train a neural to localize the interest point (bonnet or blow hole), the re-sized original image is used as input and the predicted output of the network is the masked image. The architecture of the network we used for training is shown in Figure 2.5. We call this the standard localization network.

The loss function we minimize is the categorical cross-entropy (or mutli-class logloss). The learning rate was initially set at 0.01 and reduced automatically if the validation loss does not improve after 10 epochs. The activation for each convolutional layer is ReLU [61] while the activation for the output layer is softmax. Softmax activation ensures that each pixel in the predicted mask is in the range of (0,1) and the pixels sum up to 1. In other words, it is a measure of certainty that a certain pixel in the predicted mask corresponds to the location of

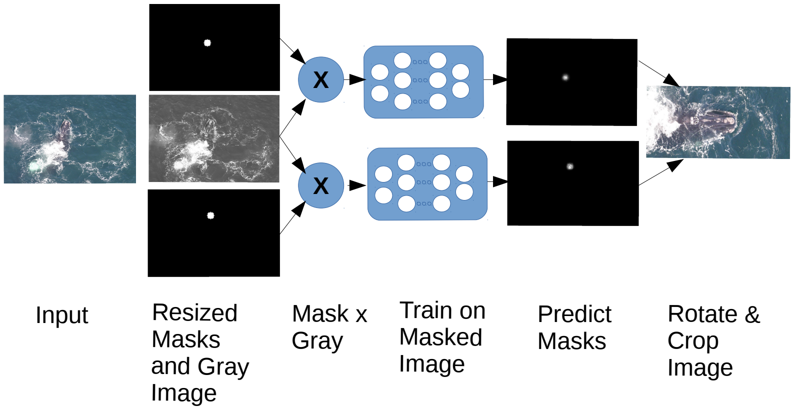

Figure 2.4: Body Localization Overview: In order to locate the body, the bonnet and blow hole masks are re-sized to 128 × 192. Also, the original image is re-sized to 128 × 192 and converted to gray scale. The two masks are multiplied with the gray scale image to produce masked images. Each of these masked images are passed into two networks to train a network to predict the location of the interest point (bonnet or blow hole). The network is used to predict the location of the interest points. These interest points are used to rotate the whale and localize the body.

the interest point. The dropout rate is 50% and the max-pooling is done over size (2,2).

Because the last layer in the network is softmax, pixels in the predicted mask are probabilities that range between 0 and 1 and that sum up to 1. It is very important to ensure that the ground truth (the labels) follow the same rules. In order to do that, we rescale the true mask (or the true labels) by dividing each pixel by the sum of the mask as shown in the Equation 2.1:

$$y_{ij} = \frac{pixel_{ij}}{\sum_{i=1}^{H} \sum_{j=1}^{W} pixel_{ij}} \tag{2.1}$$

where $pixel_{ij}$ is the pixel value at row i and column j. y_{ij} is the normalized pixel value such that $y_{ij} \in [0, 1]$ and $\sum_{i=1}^{H} \sum_{j=1}^{W} y_{ij} = 1$.

Once the two models that predict the bonnet and the blow hole are trained, they can be used to predict the locations of bonnet and the blow hole. For each mask, the pixel with the highest intensity is considered to be the one where the model predicts the interest point to be. To make the prediction more robust, we take the mean location of the top 5 pixels with the highest intensities. We found, empirically, that using the top 5 pixels achieves the best localization

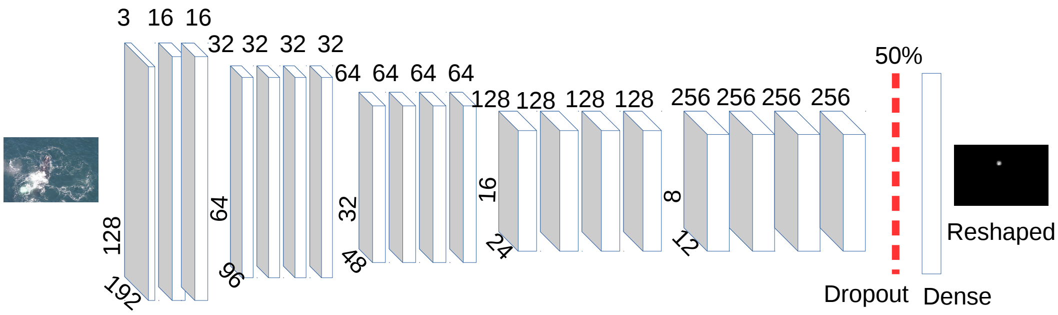

Figure 2.5: The Standard Localization architecture: the same architecture is used to localize the bonnet and blow hole, and later the head (in Section 2.4). The input of the architecture is an image of size 128 × 192. The output of the network is a layer with 128 × 192 = 24576 possible classes. The output layer is simply a flattened mask and reshaping this layer gives us back the predict mask. The pixels with the highest intensities represent the location of the interest point.

results.

To localize the body and rotate it, we first enlarge the mask from the size 128 × 192 to the original size. The top 5 pixels with the highest intensities are averaged for each of the two masks. Then, the angle of the whale with respect to the x axis is estimated by Equation 2.2.

$$\theta = tan^{-1}(\frac{y_{bonnet} - y_{blowHole}}{x_{bonnet} - x_{blowHole}}), \tag{2.2}$$

where y_{bonnet} and x_{bonnet} are the predicted coordinates of the bonnet on the y and x axises. As discussed earlier, this point is the result of averaging the coordinates of the top 5 pixels with highest intensities in the bonnet predicted mask. The same thing applies to the $y_{blowHole}$ and $x_{blowHole}$ which are the predicted coordinates of the blow hole.

The image is rotated around the estimated center of the whale head, which is given by the equation:

$$\begin{aligned} x_{headCenter} &= 0.5 \times (x_{bonnet} + x_{blowHole}) \\ y_{headCenter} &= 0.5 \times (y_{bonnet} + y_{blowHole}) \end{aligned} \tag{2.3}$$

The image is cropped from the center of the head and the size of the crop is 4 × *distance* along the x axis and $2 \times distance$ along the y axis. The *distance* value is the distance between the blow hole and the bonnet. Figure 2.10 in Section 2.6 shows a sample of images produced using the information we described in this section. As shown in Figure 2.10, the resulting images all show the whale bodies localized and pointing in the same direction.

2.4 Head Localization

Once the body is localized and rotated so that it is pointing east, we are ready to localize the head of the whale. A head mask is used to train a network to localize the head. The mask and the input image (the whale body image we produced in Section 2.3) are re-sized to size 112 (*height*)×224 (*width*). The labels (ground truth) for this network are created by multiplying (element-wise) the re-sized mask with the gray scale image.

The architecture of the network we used for training is the same one we used in the previous section (shown in Figure 2.5). The only difference is that the body size of the input image and the mask is different. Therefore, the number of parameters at each layer is different.

As shown in Figure 2.6, the whale body image is converted into gray scale. The head mask and the gray scale images are multiplied and passed to the model for training. Then, the model is trained to predict the head mask. Finally, the predicted mask is re-sized to have the same size as the whale body image. Then, the predicted mask is thresholded and converted from gray scale image into binary image. The coordinates of the largest rectangle in this binary image are used to crop the head from the body image.

We used multiple thresholding methods to convert the gray scale mask into binary mask. For instance, we thresholded the head masks using Otsu [65]. The image is thresholded as shown in Equation 2.4:

$$I(i, j)_{thresholded} = \begin{cases} 1, & \text{if } I(i, j) \geq threshold_{Otsu} \\ 0, & \text{otherwise,} \end{cases} \tag{2.4}$$

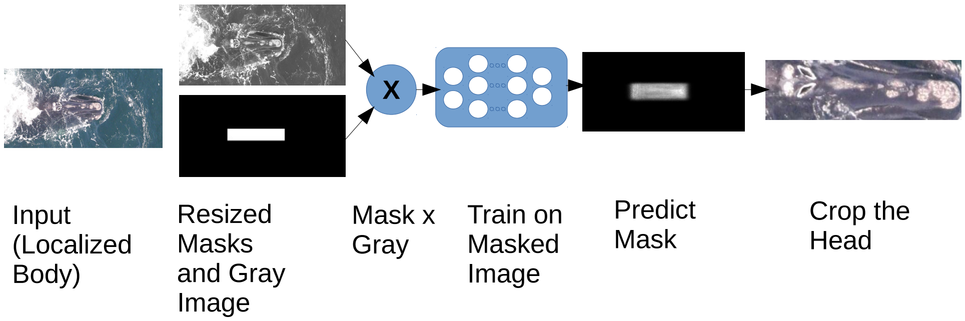

Figure 2.6: Head Localization Overview: In order to locate the head, the head mask is re-sized to 112 × 224. Also, the body image is re-sized to 112 × 224 and converted to gray scale. The mask is multiplied with the gray scale image to produce masked images. The masked image is passed into a network to train it to predict the location of the head. This network is used to predict the location of the head

In addition, we use another method to threshold the head mask according to Equation 2.5:

$$I(i, j)_{thresholded} = \begin{cases} 1, & \text{if } I(i, j) \geq \mu_{Otsu} \\ 0, & \text{otherwise,} \end{cases} \tag{2.5}$$

where μ_{Otsu} is the mean of all pixels that are higher than the ostu threshold. In other words, Equation 2.5 produces smaller head crops than Equation 2.4. During each epoch while training the recognition model (in Section 2.5), we will train on random sample from head crops produced using the Otsu method and the high mean method. We find this to be an effective data augmentation tool to reduce overfitting.

The model extracted crops of the head where all heads are normalized with respect to orientation (east) and rotation (angle = 0), translation, and scale. Figure 2.11 (in Section 2.6) shows a sample of head crops produced using the information we introduced in this section. The callosities patterns shown in Figure 2.11 are unique to each individual whale and can be used to identify a whale. These head images are passed to the recognition model (presented in Section 2.5).

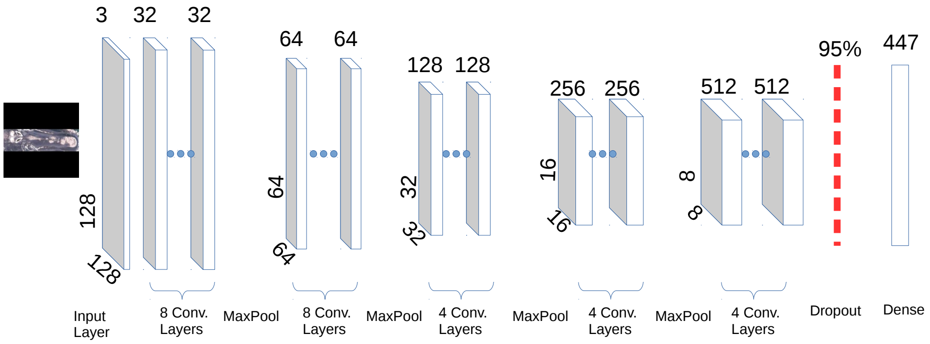

Figure 2.7: Recognition Architecture: the input to the network is an image showing the callosities of the whale. The output size is 447 corresponding to 447 unique whale IDs.

2.5 Recognition

Now that the head is extracted from the original image, it is time to train a model to recognize the individual whales. Right whales can be recognized by the callosities on the top of their heads. It is estimated that there are 450-500 north Atlantic whales remaining. However, the dataset only contains 447 individual whales. The network we trained can predict the ID of the whale by examining the callosities. This is very similar to the face recognition problem where the ID (or name) of the person is recognized by examining the facial features.

The network we used for training is described in Figure 2.7. Driven by the success of the VGG architecture [75], we opted for a similar architecture where the convolutional filter is small (3,3) and the network is very deep. The small convolutional filter helps in regularizing the network because each neuron is connected to a small number of neurons in the previous layer. We did not use any fully connected except for the output layer. Normally, the dropout rate is set at 50%. However, for this problem we set the dropout rate to a relatively high value which is 95%. We noticed that setting this value to lower than 75% leads to overfitting after only 10-20 epochs.

The image is padded by 1 pixel to ensure that the spatial resolution does not decrease except after the pooling layer. We used a (2,2) max pooling to sub-sample the response and detect more abstract features. The activation function we used is the leaky rectification (with leaki-

Table 2.1: Data augmentation: Random transformations along with parameters. These transformations are applied randomly to each image before sending it to the GPU.

Transformation	Parameters
Rotation	Angle between -20 and +20
Horizontal Flip	Randomness=50%
Vertical Flip	Randomness=50%
Horizontal Shift	Up to 12 pixels
Vertical Shift	Up to 12 pixels
Gaussian Blurring	Up to $\sigma = 1$
Contrast Rescaling	Randomly stretch/shrink intensity

ness=0.3) [58], [26] which ensures that the gradient is not 0 for negative pre-activation. The activation in the output layer is softmax which ensures that the output produced is a probability (between 0 and 1, and all classes sum up to 1). All layers were initialized randomly.

It is worth mentioning that there is some variation in the aspect ratio of the head images. However, the network expects all input images to be of the same size (in our case, it is 256×256). In order to avoid distorting the callosities image when resizing the image, we pad the image so that the image size becomes $width \times width$.

Given the small size of the training set, it is essential to augment the data to alleviate overfitting. Table 2.1 shows the list of random augmentation we used along with their parameters.

The network was trained for 520 epochs. During each epoch, a random sample of training images is created from different sources. For instance, for each epoch we randomly choose head crops images which were created using the Otsu thresholding method. In addition, we combine them with randomly chosen head crops created using the high mean method (described in Section 2.4).

The learning rate was initially set at a very low value 0.003. While training, the validation is continuously being evaluated. If the validation loss does not improve after 10 epochs, the learning rate is automatically reduced by 50%.

2.6 Implementation and Results

In this section, we describe how the model was implemented and the results. The data is hosted on Kaggle [43]. Kaggle is the platform where the data is stored in. Once a solution is submitted to Kaggle, the server will evaluate the solution and rank it against other solutions. The size of the data is very small with respect to the number of labels. The size of the training set is 4544 images. For the training set, both the images and the labels (individual whales IDs) are provided. On the other hand, for the testing set, only the images are provided without the labels and the size of this set is 6,925 images. In order to perform validation locally, we extract a validation set from the original training set. The size of the validation set is 10% of the training set (452 images). Therefore, the training set size is reduced to 4092.

The number of whales in each individual whale varies significantly from whales with 1 training image up to 47 training images. Figure 2.8 shows a summary of the number of whales with a certain number of images. For instance, there are 24 whales (or classes) with only 1 training image and 29 whales with 2 training images. On the other hand, there is 1 whale with 47 training images. The average number of training images in each class is 10 training images.

We developed our model on a laptop equipped with GTX980M (4GB) graphics card. Only 3.5GB of GPU RAM was available for training. The code was developed using Theano [2, 4] which is a python library for optimizing and evaluating mathematical expressions in multidimensional arrays. We also used keras [9] which is a highly modular library to train neural networks on GPUs or CPUs.

In order to pre-process the image data and to perform geometric transformations, we used scikit-image [83] and openCV [6]. The networks were trained in batches of size 32 images. This is the largest batch size we could fit in the GPU memory. The CPU performs data augmentation on each batch before sending it to the GPU for training.

Training each of the three localization networks took around 9 hours each. On the other hand, training the recognition network took around 50 hours. Therefore, the total training time for all the networks is 9 + 9 + 9 + 50 = 77 hours.

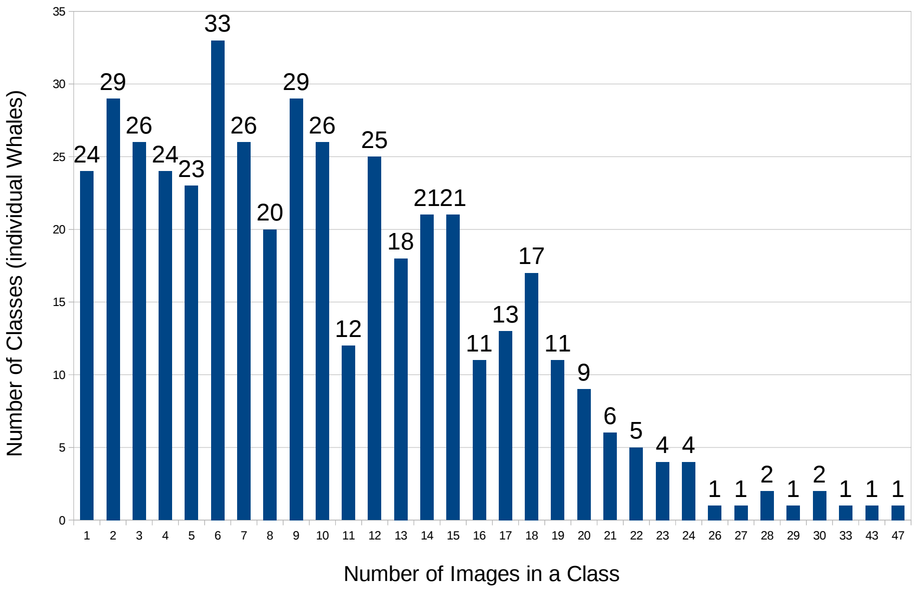

Figure 2.8: A summary of the number of whales with a certain number of images. There are 24 whales (or classes) with only 1 training image and 29 whales with 2 images. On the other end of the chart, we can see that there are few classes with a relatively high number of training images. For instance, there is one whale with 47 training images.

The metric used by the server to evaluate the predictions is the multi-class logarithmic loss (also known as categorical cross-entropy). The equation for this metric is:

$$logloss = -\frac{1}{n}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}log(p_{ij}) \tag{2.6}$$

where N is the number of images in the predictions file (or the number of images in the test set), M is the number of labels (the total number of individual whales). y_{ij} is a mapping from the image i to the true label j (for example, y_{ij} is 1 if the image i belongs to the whale j and 0 if it does not). $log(p_{ij})$ is the natural log of the predicted probability made by the model that the image i belongs to whale j.

While training the recognition network, we minimize this metric directly. Figure 2.9 shows this loss function as it is being minimized during training. The log loss goes as low as 1.62 at the end of the training.

Table 2.2: Teams Ranking: this table shows the ranking of the solution we describe in the paper. The solution ranked in the $5^{th} position$. The table only shows the top 10 teams while the number of teams that participated in the competition is 364. The full table can be found on the web page of the competition [43]

Ranking	Team Name	Score
1	deepsense.io	0.59600
2	felixlaumon	1.07585
3	SKE	1.14982
4	threedB	1.33648
5	AbdulWahab	1.46909
6	Tsakalis Kostas	1.51900
7	bawdyb .	1.55823
8	Left Whales	1.75764
9	Anil Thomas	1.80178
10	Doug Koch	2.13797

On the server, the log loss score we achieved is 1.47. The top-1 accuracy of the module on the validation set is 69.7% while the top-5 predictions accuracy is 85.0%.

We used the same loss function to train the localization modules. Figure 2.9 show the train and validation log loss progress for both the bonnet localization network and the blow hole network, respectively. In Figure 2.9, we can see that the loss function of the bonnet localization network decreases from 4.5 to around 2.28 at the end of training.

As shown in Figure 2.9, the blow hole localization loss decreases from 5 to around 2.52 at the end of the training. The performance of the bonnet network is slightly better than the blow hole localization network as the former has a lower loss than the latter. This is likely because for many samples the blow hole is completely covered by water (white pixels) while the bonnet is visible in most of the images.

Figure 2.9 also shows the performance of loss function of the head localization network. The loss value goes down from 5.03 to around 3.87. Because both the validation and training losses curves are very close to each other, the performance may be improved slightly by using a larger network.

In the context of localization, the log loss function may be difficult to interpret. Of course, the lower the loss function, the better we can expect the localization to be. However, it is also

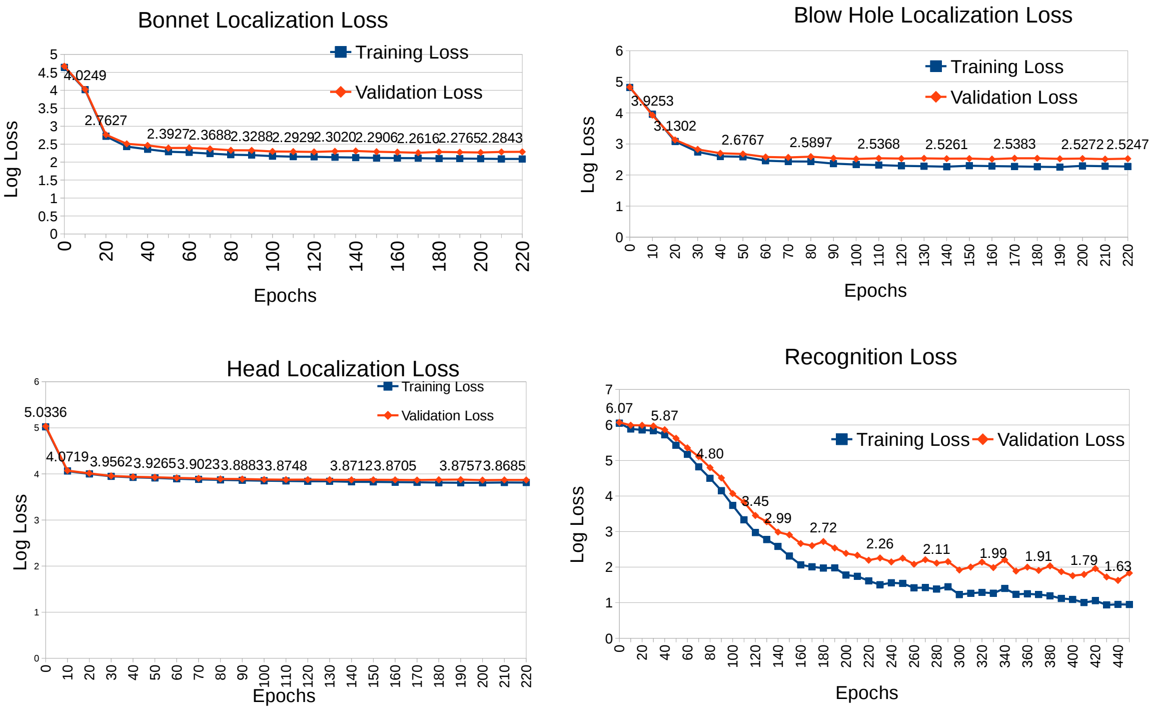

Figure 2.9: Loss Curve: This figure shows the training and validation loss during the training of the four deep learning models (bonnet localization model, blow hole model, head localization model, and whale recognition model.)

useful to track the quality of the localization visually. Figures 2.10 and 2.11 show a random set of images of localized whales bodies and localized whale heads, respectively.

In addition, Figure 2.12 shows the bonnet location prediction after every 5 epochs (up to epoch 35). The upper row shows the true location of the bonnet while the lower row shows the prediction made by the network. Because we augment the data by flipping the image horizontally and vertically, the true location of the bonnet (in the upper row) changes. At the beginning of the training (epoch 0), the network predicts the bonnet to be in the middle. Then, the prediction starts to improve gradually. At epoch 20, we start to see the network correctly tracking the location of the bonnet. Of course, monitoring one image is not enough so we monitor a small sample of images. Once we were satisfied with the performance of the localization network, we stopped the training. Figure 2.12 shows one of the monitored images while training the bonnet localization network. We monitor the training performance in the

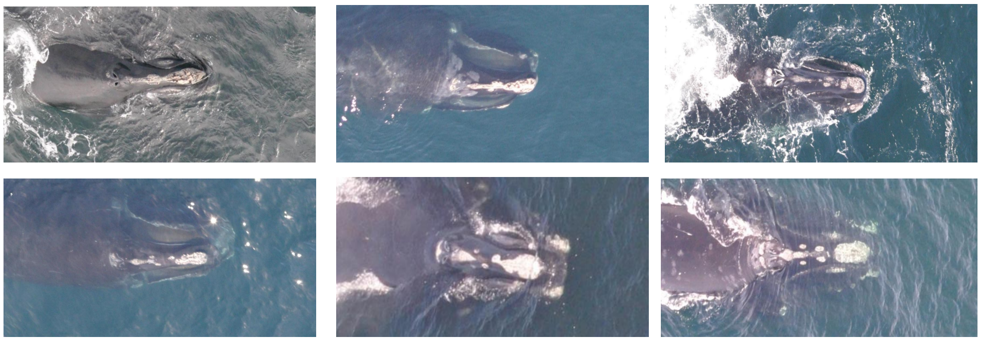

Figure 2.10: A sample of localized whale bodies.

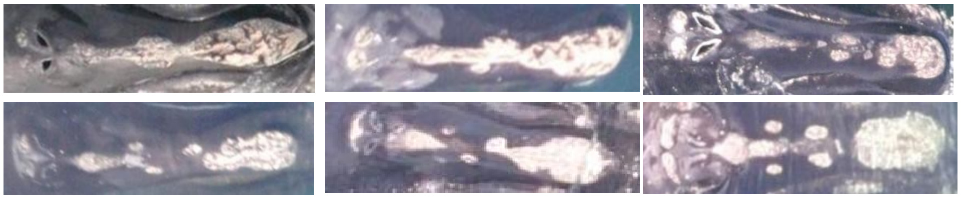

Figure 2.11: A sample of localized whale heads.

same manner when training the blow hole and head localization networks.

As we mentioned earlier, using a multi-stage approach was the only way to be able to train a deep learning model on this data set. However, when carrying out a multi-stage approach, there is a risk of error propagating from one stage to the next. Figure 2.13 shows a sample of cases where the whale body crops could not be localized correctly. In the images shown in this Figure, some of them do not show the full callosities pattern on the whale head while other include the whale body oriented in the wrong directory. There are approximately 0.7% of cases where the model could not correctly localize the body.

The head localization error is higher than the body localization error. The head localization error is 2.9%. Figure 2.14 shows a sample of images where the head was not localized correctly. These images include cases where the callosities pattern is not fully shown in the image.

While the head localization error is relatively low, the intersection over union metric show that there is a room for improvement. The intersection over union is defined as shown in

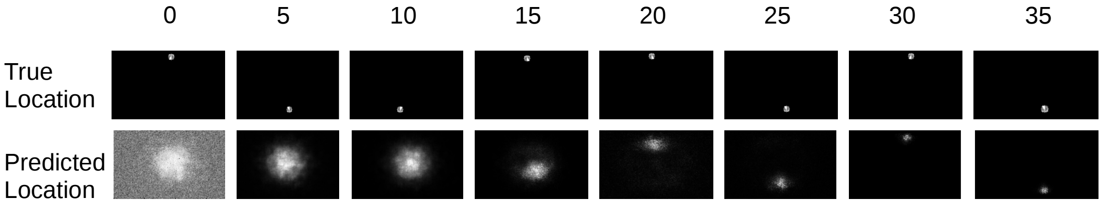

Figure 2.12: This figure shows how the network performance in tracking the location of the bonnet improves as it is being trained. At epoch 0, the network predicts the location of the bonnet to be in the middle of the image. Later, the network gradually becomes capable at predicting the location of the bonnet.

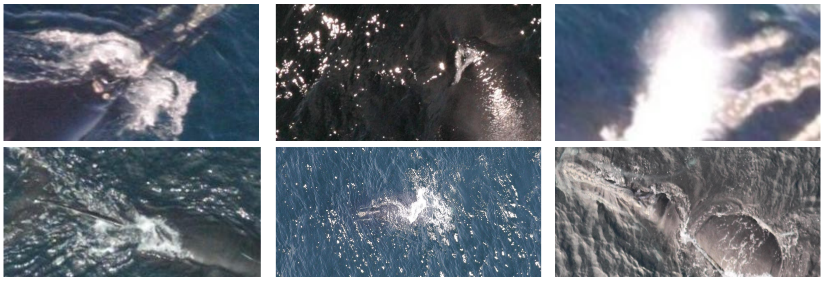

Figure 2.13: A sample showing cases where the body of the whale was not localized correctly. In the images shown in this Figure, some of them do not show the full callosities pattern on the whale head while other include the whale body oriented in the wrong directory.

Equation 2.7:

$$IoU = \frac{M_{true} \cap M_{pred}}{M_{true} \cup M_{pred}} \tag{2.7}$$

where M_{true}, M_{pred} are the true and predicted masks. On the validation set, the average body localization IoU is 0.51. In other words, while the percentage of complete failure in localizing the whale head is relatively low (2.9%), the quality of the head localization can be improved. We strongly believe that using a better model to perform head and body localization is likely to yield better IoU score. A lower IoU score is likely going to lead to a lower whale classification error.

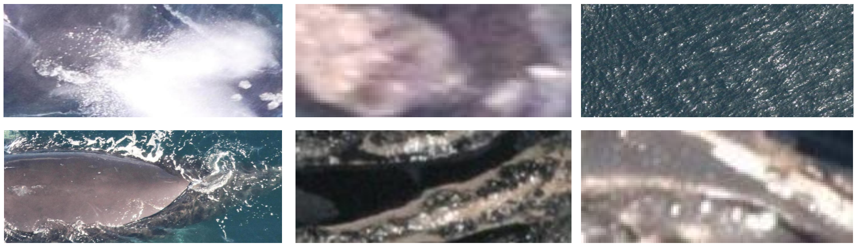

Figure 2.14: A sample showing cases where the head of the whale was not localized correctly. These images include cases where the callosities pattern is not fully shown in the image.

2.7 CONCLUSION

We introduced a method to recognize individual whales from the callosities on their head. This method can help in the conservation efforts of marine biologists. Because the size of the available training images is very low, overfitting is very difficult to avoid. In fact, we could not achieve an accuracy of more than 8% when training a deep learning model to classify the whales directly. We solved this problem by introducing a model to localize the head of the whale and training the recognition model on it. This helps the recognition model to focus on the callosities features located on the head of the whale. Our model's top-1 and top-5 predictions accuracies are 69.7% and 85%, respectively. We strongly believe that the performance can be boosted by increasing the size of the training set. In addition, improving the body and head localization models will likely improve the whale classification error.

In the next chapter, we will improve on the current results. The current accuracy on the validation set is 69.7%. We will improve this score to 78.7%.

Chapter 3

Right whale Recognition using the Wide Localization Network

3.1 Introduction

In the previous chapter, we described how the Right whale recognition problem can be solved by localizing the bonnet and blow hole using the standard localization network. In this chapter, we will improve the results by using the wide localization network. Furthermore, after orienting the heads towards the east, we will localize three points on the of the whale. Then, these points will be used to slightly rotate the head to make sure that the angle with the x axis is exactly 0. In addition, these three points will be used to extract the region of interest.

3.2 Method Overview

Training the recognition model on the raw images is unlikely to yield good results. When training the recognition model on the original images, we could not achieve good results due

---

[0] The majority of this chapter was originally published in: AbdulWahab Kabani, and Mahmoud R. El-Sakka "Improving Right whale Recognition by Fine-tuning Alignment and Using Wide Localization Network", Canadian Conference on Electrical and Computer Engineering (CCECE), Windsor, Canada, May 2017 [pre-print]. [40]

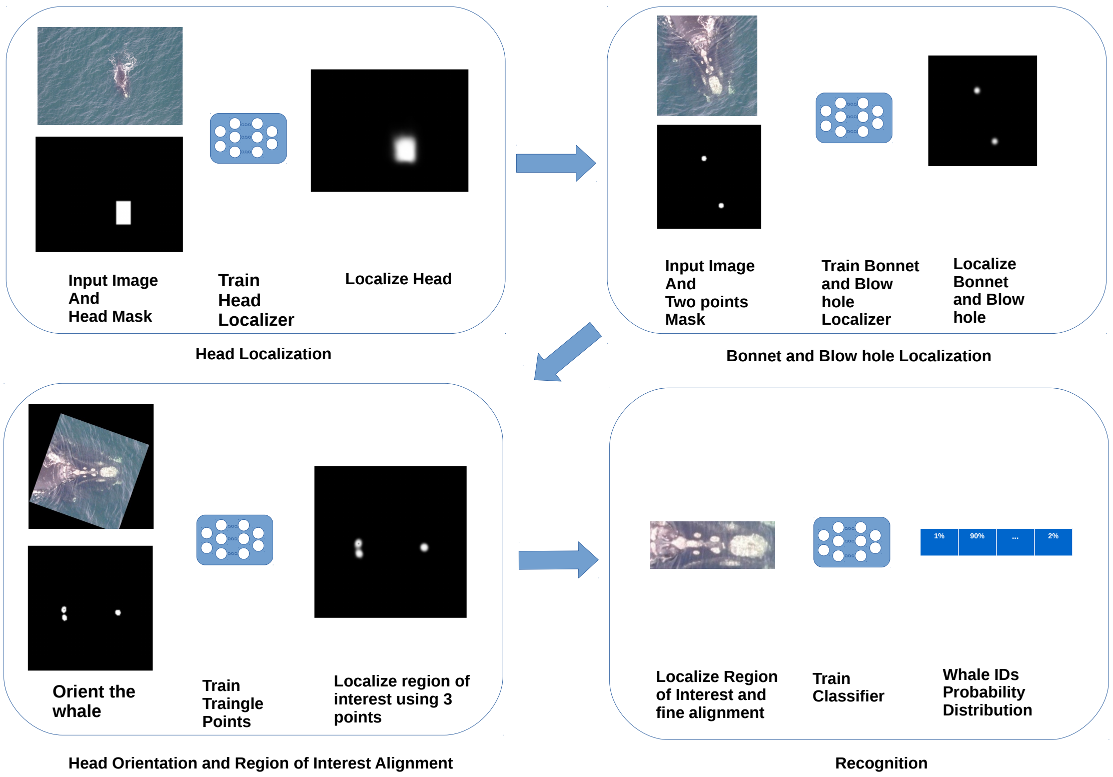

Figure 3.1: The overview of the method: the first stage involves performing head localization. The second stage involves identifying the bonnet and blow hole. Using these two points, the head is oriented east. In stage 3, we perform further fine tuning alignment and a bounding box is extracted. Once the region of interest is identified, the region of interest is passed to the recognition network to predict the ID based on the callosities pattern.

to overfitting. The max accuracy we were able to achieve is around 8%. Since the number of target labels is large (447) and the number of training data is small (4,544), the recognition model will pick up on non-discriminative features from the surrounding water. In addition, when resizing the images to fit the model, the most discriminative features (head callosities) will be very small making it very difficult to recognize the whale. To avoid this, we need to produce head patches that show only the head callosities with as little background as possible.

The summary of this method is shown in Figure 3.1. First, we start by training a wide convolutional neural network (wide net) to localize the head of the whale. Once the network is trained, a bounding box is predicted and used to produce a head image. This is similar to what was suggested in [5]. However, we used a different localization network called wide net. The main advantage of this network is that it has the ability to predict region of interests with different shapes (rectangles, circles, or any arbitrary shaped region of interest). In addition, it has a low memory requirement making it ideal for use as a pre-processing step.

In the second stage, a wide net model is trained to localize the bonnet and blow hole. As suggested in [82], we use these two points to orient the head in one direction. In the third stage, we train a wide net model to predict three points (bonnet and two points representing the post blow hole callosities). These points are used to perform further fine-tuning alignment. In addition, we can use them to extract a very tight bounding box that only shows the most discriminative features.

Once a bounding box is extracted, it is used to train an ensemble of deep learning models to classify and identify each image. What is unique about our solution is that we use deep learning model called the wide net, which we will describe in Section 3.3. In addition, we perform an extra fine-tuning step after head orientation using three support points.

3.3 Head Localization

In order to localize the whale head, we propose an efficient deep learning model called the wide convolutional neural network (wide net). The architecture of this model is shown in Figure 3.2. This network is capable of performing segmentation and producing region of interests at a very low processing power. The network is composed of three levels: Level_1 is simply a set of convolutional and pooling layers. Level_2 involves merging the last layer from each block in level_1 in three different ways. Three types of merging is done: type_1 involves merging (block_2, block_3, block_4, block_5), type_2 involves merging (block_3, block_4, block_5), and finally type_3 (merging block_4, and block_5). Type_1 carries a combination of high resolution features along with highly abstract features. On the other hand, Type_3 carries only abstract features. Finally, in level_3, the outputs of level_2 are combined and a mask is produced.

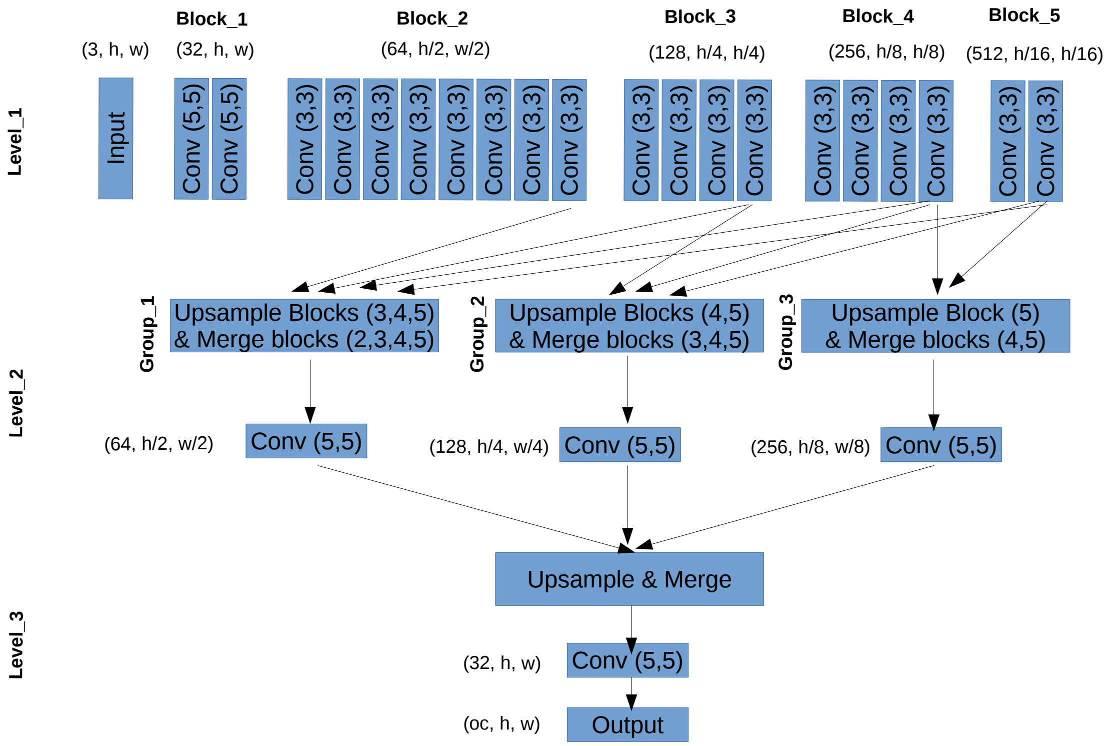

Figure 3.2: The architecture of wide net: the network is composed of three levels. Level_1 is just like any typical recognition network. Level_2 involves merging the last layer from each block in level_1 in three different ways. Finally, in level_3, the outputs of level_2 are combined and a mask is produced. Abbreviations: h is the height of the image, w is the width of the image, oc is the output channel size.

The last layer in the network has a sigmoid activation to map the pre-activation values into values between 0 and 1. Rectified linear units (ReLU) activation [61, 30] is used after each layer. Each pre-activation is followed by batch normalization [36]. In order to extract more abstract features, we use Max-pooling in level_1. As a result, the layers in each block in level_1 will have different sizes. In order to combine them together in level_2, we use upsampling to ensuring that the layers can be merged. This network has a low processing footprint and was previously tested on a device with only 3.5 GPU memory.

Before passing the image to the network, it is important to standardize the images. We opted for channel-wise mean subtraction and standard deviation division. All images were resized to $128(height) \times 192(width)$. The ground truth for this network involves a mask that contains zero pixels everywhere except in the region of interest.

Because we are using a sigmoid activation in the output layer, our loss function is binary cross-entropy. The learning rate was set at 0.001 and decreased by 50% if the validation error does improve after 20 epochs. The head localization model is trained for 100 epochs. After that, we use it to predict the bounding box for the whale head. We are now ready for the bonnet and blow hole localization.

3.4 Bonnet and Blow Hole Localization

The bonnet and blow hole localization stage (shown in Figure 3.1) involves training a deep learning model to locate the two points. This was proposed in [82] as a means to speed up the training process. We also noticed that it is useful in reducing the amount of pixels that represent the surrounding water. These pixels can lead to overfitting because the training set is not large enough. Therefore, the model may be tricked into thinking that these pixels influence the kind of target we are interested in.

The training process and network architecture (Figure 3.2) is exactly the same as we described in Section 3.3. However, the input images were resized into $128(height) \times 128(width)$. In addition, this network will output a mask with 2 channels. One channel produces the location of the bonnet while the other channel predicts the location of the blow hole.

Once these two points are localized, we can easily rotate the images. This is because the angle of the whale with respect to the x axis can be estimated by Equation 3.1.

$$\theta = tan^{-1}(\frac{y_{bonnet} - y_{blowHole}}{x_{bonnet} - x_{blowHole}}), \tag{3.1}$$

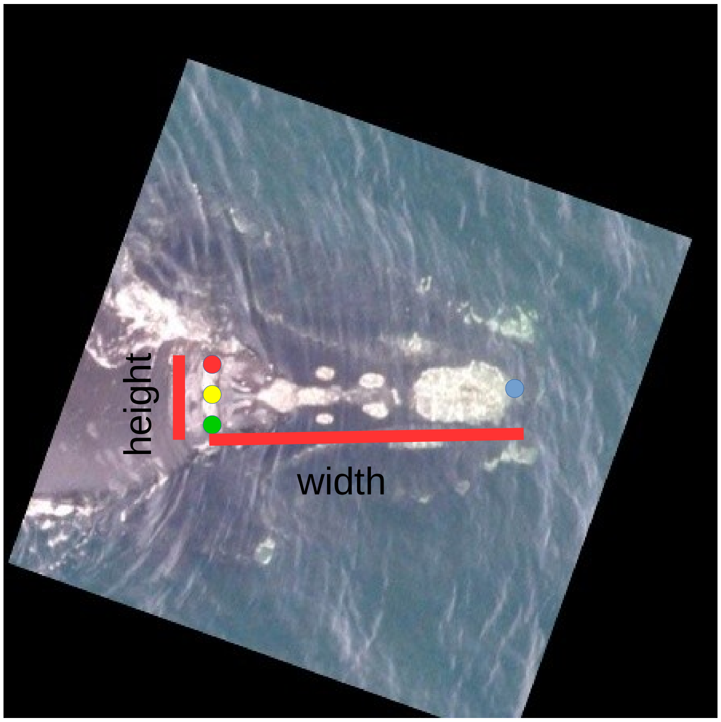

Figure 3.3: Three points are detected: the bonnet and two points representing the post blow hole callosities. The post blow hole callosities points can be used to extract the height of the region of interest. The bonnet and the average of the post blow hole callosities points are used to get the width and perform fine alignment by rotating the image if the angle with the horizon is more than 1 degree.

3.5 Head Orientation and Region of Interest Alignment

After orienting all whale heads into one direction, we train the same localizer network described in Section 3.3 and 3.4 to detect three points (the bonnet and the two points that represent the post blow hole callosities). We perform this step for two main reasons. First, these points can be used to perform even more fine tuning alignment than we achieved in the previous step. Second, these points can help in extracting very small region of interest which is what we are trying to achieve. The average distance between the bonnet and the two post blow hole callosities is the width of the region of interest. The distance between the two post callosities

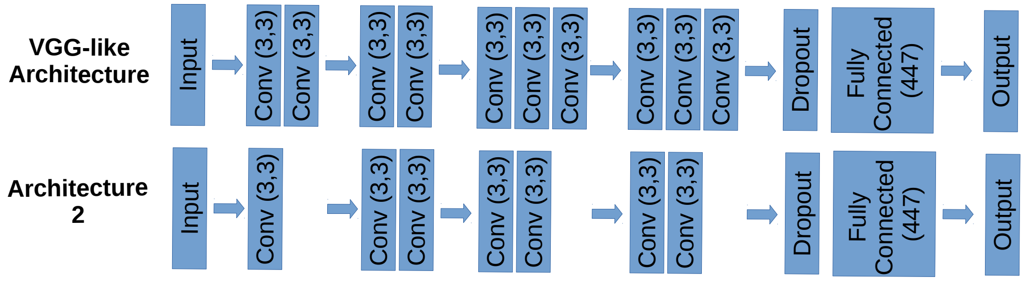

Figure 3.4: Recognition Architectures: we use two architectures. One architecture is a VGG-like structure [75] but without the fully connected layers. The second architecture is a simpler architecture with less layers.

points is the width of the image.

3.6 Recognition

As mentioned earlier, Right whales can be recognized using the callosities patterns on their heads. We used an ensemble of two architectures (a VGG-like [75] structure without the fully connected layers and a simpler architecture). These two architectures are shown in Figure 3.4. The ensemble includes training these two networks with three kinds of non-linearities: leaky non-linearity [58] with leakiness value of 0.1, leaky non-linearity [58] with leakiness 0.3, and a rectified linear units [61]. We set the dropout rate [33] at 75%. All layers were initialized randomly. Table 3.1 shows the kind of data augmentation that we used to alleviate overfitting. All networks were trained for 400 epochs, if the validation error does not improve after 25 epochs, it is automatically decreased by 50%.

3.7 Results

As mentioned earlier, the training set has 4,544 images while the testing set has 6,925 images. We extracted 10% of the training set and create a validation set so that we can monitor the

Table 3.1: Recognition Data augmentation: Random transformations along with parameters. These transformations are applied randomly to each image before sending it to the GPU.

Transformation	Parameters
Rotation	Angle between -10 and +10
Horizontal Flip	Randomness=50%
Vertical Flip	Randomness=50%
Horizontal Shift	Up to 19 pixels
Vertical Shift	Up to 12 pixels
Gaussian Blurring	Up to $\sigma = 1$
Contrast Rescaling	Randomly stretch/shrink intensity

training process. The number of images for each class varies from 1 training image per class to 47 training images per class. There are 24 classes with only 1 training image. In order to process the images, we used scikit-image [83] and openCV [6]. The deep learning models were developed using Theano [2, 4] and Keras [9].

Figure 3.5 shows the sample output of each stage. Training the localization networks took around 5 hours per network. Training the recognition networks took around 3 hours per network. We trained an ensemble of 12 networks. Table 3.2 shows a summary of the results that we achieved. The head localization model achieved an intersection over union score (IoU) of 71.8%. The average absolute distance error from the center of the true bounding box to the predicted one is 2.41 pixels. The log loss score (lower is better) of the recognition ensemble is 1.16 on the validation set and 1.10 on the test set as reported by the evaluation server. The accuracy score on the validation set is 78.7%. We cannot compute the accuracy on the test set because the ground truth is not provided. Table 3.3 shows how our method stacks against other solutions. The table only shows the top 10 teams out of 364 teams. The solution we described in this paper ranks in the top 3 positions. It is worth noting that the solution that was ranked in the fifth position is our old method which we described in Chapter 2.

Table 3.2: Results of all Models we described in the chapter. All results are reported on the validation set unless mentioned otherwise. The test set log loss is reported as displayed by the evaluation server. Because we do not have the ground truth, we cannot report the accuracy.

Model	IoU	Average Distance (in pixels)	Log Loss	Accuracy
Head Localization	71.8%	2.41	_	_
Bonnet and Blow hole	_	(bonnet: 2.02, blow hole: 2.64)	_	_
Bonnet and Two Post-callosities Points	_	(Bonnet: 1.26 Point_1: 2.01 Point_2:1.90)	_	_
Recognition (Validation Set)	_	_	1.16	78.7%
Recognition (Test Set)	_	_	1.10	Unknown

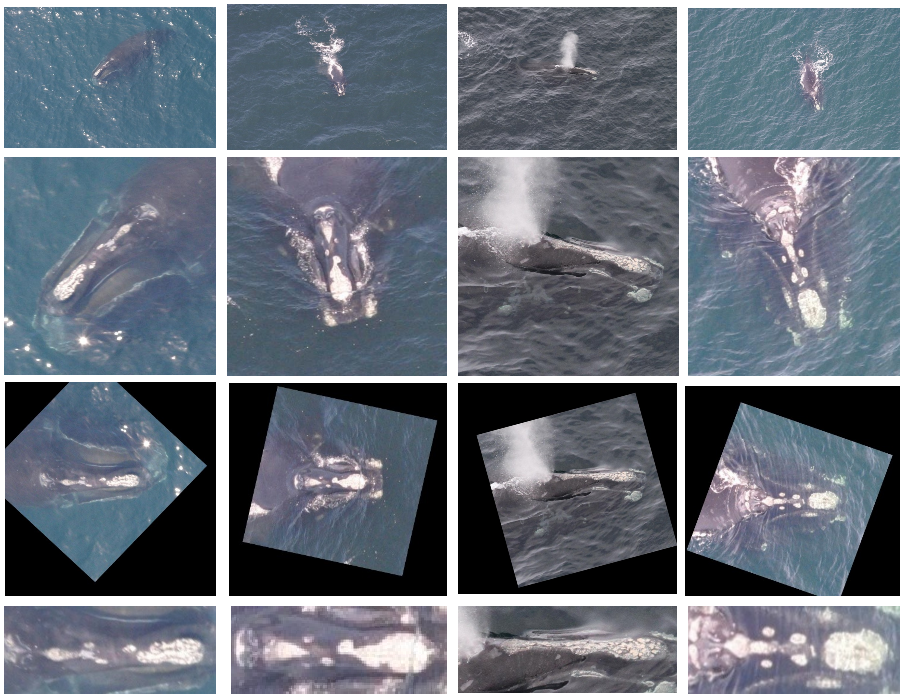

Figure 3.5: A sample of input images along with the localization after each stage.

3.8 Conclusion

Deep learning can be used on problems with large training data. We proposed a wide localization network which can be used to localize a region of interest in an image. We demonstrated how it can be used to localize the callosities pattern. After that, we trained an ensemble of deep learning models to classify the images.

In this chapter, we improved on the results we achieved in Chapter 2. In the next chapter, we will study a different problem, which involves estimating the left ventricle volume and ejection fraction. We will follow a similar approach where we localize the region of interest in the image. In Chapter 4, we present a method based on the standard localization network. A better solution based on the wide localization network is presented in Chapter 5.

Table 3.3: Teams Ranking: this table shows the ranking of the solution we describe in the paper. The solution ranked in the $3^{rd} position$. The table only shows the top 10 teams while the number of teams that participated in the competition is 364. The team that ranked in the fifth position is our old solution which we described in Chapter 2.

Ranking	Team Name	Score
1	deepsense.io [5]	0.59600
2	felixlaumon [49]	1.07585
Post_competition	**ours**	**1.1025**
3	SKE	1.14982
4	threedB	1.33648
5	AbdulWahab (Chapter 2)	1.46909
6	Tsakalis Kostas	1.51900
7	bawdyb .	1.55823
8	Left Whales	1.75764
9	Anil Thomas [82]	1.80178
10	Doug Koch	2.13797

Chapter 4

Heart Volume Fraction Prediction Using Standard Net

4.1 Introduction

In this chapter we will look into using the standard localization network in estimating the volume and ejection fraction of the left ventricle. Cardiologists can assess cardiac function by analyzing the end-systolic and end-diastolic volumes, and ejection fraction. These values can be manually measured by a cardiologist but the process is slow and time consuming. We introduce an automated method that can estimate these values. We developed our method on a data set with 500 training studies, 200 validation studies and tested it on 440 testing studies. The data set is compiled by the National Institutes of Health and Children's National Medical Center [41].

In general, left ventricle volume can be estimated by segmenting the left ventricle cavity [55, 57, 20, 44]. Once the cavity (the blood pool) is segmented, the volume can be calculated by summing up the sub-volumes of all slices according to Simpson's rule.

---

[0] The majority of this chapter was originally published in: AbdulWahab Kabani, and Mahmoud R. El-Sakka "Estimating Ejection Fraction and Left Ventricle Volume using Deep Convolutional Networks", International Conference on Image Analysis and Recognition (ICIAR), Pvoa de Varzim, Portugal, July 2016. [37]

In this paper, we take a slightly different approach and estimate the volume using a convolutional neural network. A Convolutional Neural Network (convnet or CNN) is a special type of neural network that contains some layers with restricted connectivity. CNNs have been producing excellent results on many classification tasks. This is all thanks to the availability of large training data sets [11, 73] , powerful hardware, regularization techniques such as Dropout [33, 77], initialization methods [23], ReLU activations [61], and data augmentation. Since 2012, many networks that can perform classification were introduced [48], [79], [75].

In general, CNNs can produce excellent results on many classification tasks if large amounts of training data is available. However, this requirement can be alleviated if the data is cleaned, standardized, and transformed such that to minimize viewpoint variance. The total size of the data set described in this paper is 1140 studies (500 training studies, 200 validation studies and 440 testing studies). On this dataset and with only 500 training studies, we were able to estimate the volume with around +/- 15 ml error. Furthermore, we were able to estimate the ejection fraction with +/- 5% error. The ejection fraction is one of the most important cardiac measurements since it describes how good the heart is in pumping blood.

First, we localize the left ventricle (Section 4.2). Then, the data is preprocessed, cleaned and standardized (Section 4.3). Finally, the volume of the left ventricle is estimated (Section 4.4). In Section 4.5, we present the results. We conclude our work in Section 4.6.

4.2 Left Ventricle Localization

For each patient, there are 8-16 short axis views (slices). These views show cross-sections of the left ventricle at different levels. Each slice contains around 30 images spanning the cardiac cycle (one heartbeat). The end-diastolic volume is the volume when the left ventricle is fully expanded while the end-systolic volume is when the left ventricle is contracted. To estimate these two volumes, for each slice, we need to identify the image with the largest blood pool area (end-diastole) and the one with the smallest area (end-systole).

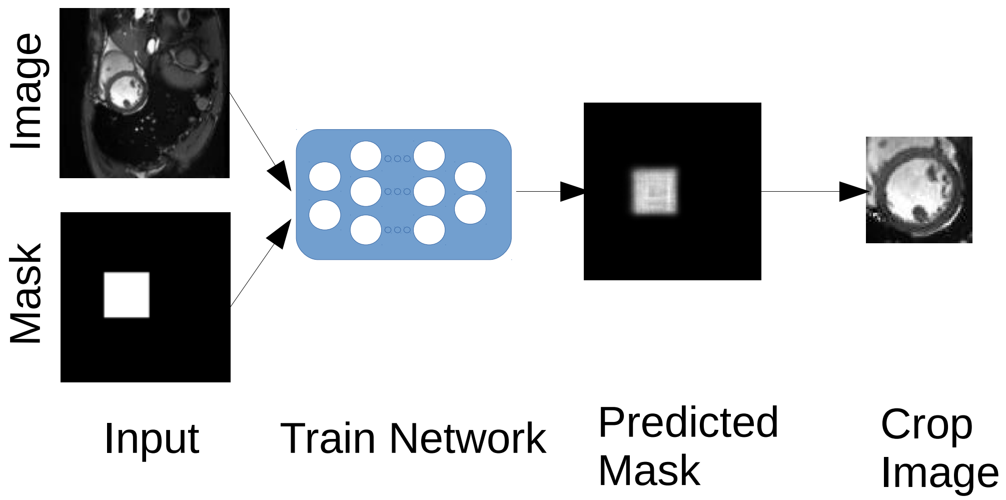

Figure 4.1: Model Overview: A set of training images along with masks are used to train the network. Once the network is trained, it can be used to predict a mask which identifies the location of the region of interest. Finally, this predicted mask is thresholded (using Otsu [65]) and used to crop the image.

In order to do that, we first localize the left ventricle and crop the image to get rid of the background pixels. We train a localization network on the training images and on the corresponding masks. The masks indicate where the left ventricle is in the image. Figure 4.1 shows the summary of the localization.

The architecture of the localization network (shown in Figure 4.2) is similar to many classification networks. The image goes through many convolutional and maxpooling layers. The output layer has $h \times w$ number of units where h and w are the dimensions of the input image. Reshaping the output layer produces the predicted mask. Once thresholded, the predicted mask can be used to predict the location of the left ventricle. In the predicted mask, the pixels with high intensity values correspond to where the network predicts the left ventricle to be whereas pixels with values close to 0 correspond to background pixels.

Each convolutional layer is followed by ReLU activation [61]. The output layer has a softmax activation to ensure that the sum of all pixels in the predicted mask is 1 and that the value of one pixel is between 0 and 1. Maxpooling is used to detect features at different scales. The network is regularized with a 50% dropout rate.

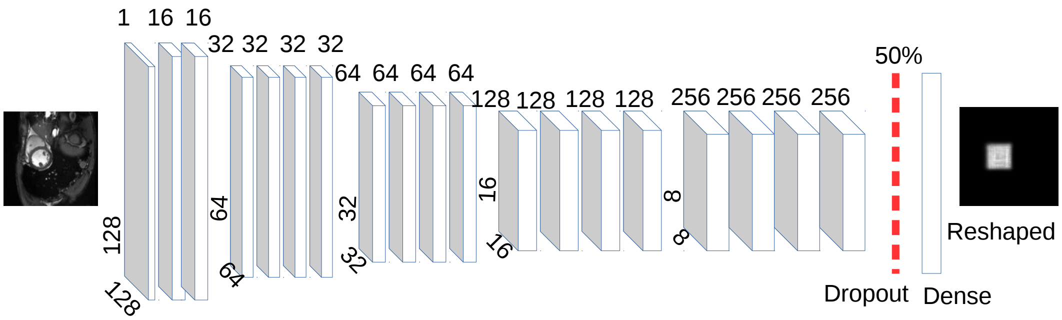

Figure 4.2: Localization architecture: The input of the architecture is an image of size 128 × 128. In this figure, the output of the network is a layer with $128 \times 128 = 16,384$ possible classes. The output layer is simply a flattened mask and reshaping this layer gives us back the predicted mask. The pixels with the highest intensities represent the location of the region of interest.

The pixels of the input mask are supposed to be probabilities (sum up to 1 and their range is between 0 and 1). Therefore, we standardize each pixel in the input mask by Equation 4.1:

$$y_{ij} = \frac{pixel_{ij}}{\sum_{i=1}^{H} \sum_{j=1}^{W} pixel_{ij}} \tag{4.1}$$

where y_{ij} is the normalized pixel value such that $y_{ij} \in [0, 1]$ and $\sum_{i=1}^{H} \sum_{j=1}^{W} y_{ij}$ sums up to 1, $pixel_{ij}$ is the pixel value at row i and column j.

4.3 Preprocessing

Once the left ventricle is localized, the meta fields are used to standardize and clean the data. First, we re-size all images using the pixel spacing and slice location meta fields. The images in this data set have different pixel spacing. This ensures that all images sizes are measured in the same units despite the difference in pixel spacing. Equations 4.2 and 4.3 ensure that the volume of the blood pool inside the left ventricle is directly proportional its area inside the image.

$$w_{new} = w_{old} \times p_w \times \frac{\sqrt{\Delta s}}{\sqrt{10}} \tag{4.2}$$

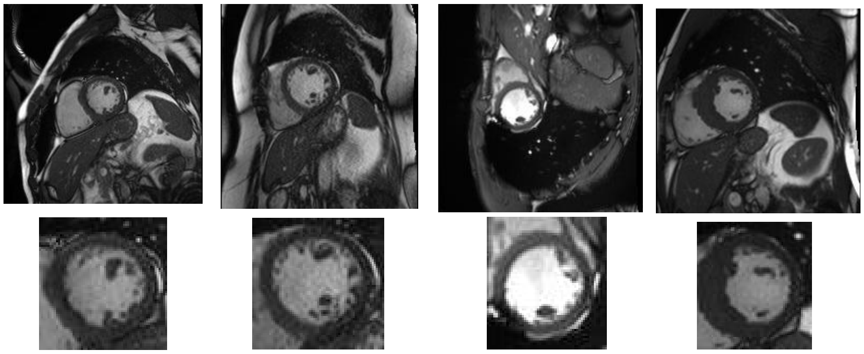

Figure 4.3: A random sample of MRI images showing the heart. In the lower row, the left ventricle is localized and cropped.

$$h_{new} = h_{old} \times p_h \times \frac{\sqrt{\Delta s}}{\sqrt{10}} \tag{4.3}$$

where w_{old} is the old image width, w_{new} is the new image width, h_{old} is the old image height, h_{new} is the new image height, respectively. p_w and p_h are the pixel spacing for the width and height. Δs is the slice height. Since most slice heights in the data set are 10 mm, we normalize each equation by $\sqrt{10}$.

Before passing the slices to the neural network to estimate the volume, the slices needs to be standardized. Many slices are in arbitrary order in the study. Therefore, we use the meta field *slicelocation* to sort all slices. This sorted almost all slices from the top of the heart to the bottom. However, since the field *slicelocation* is based on an unkown reference point, for some studies the slices were reversed (from the bottom of the heart to the top). Therefore, we trained a small neural network (Figure 4.4) to predict the slice location. We mainly use this network to predict the top and bottom slices of the heart, if these appear to be in the wrong order, we reverse them.

Figure 4.5 shows the end-diastolic image from each slice in subjects 1 and 12. The slices of subject 1 are in the correct order (from the top of the heart to the bottom). On the other hand, the slices of subject 12 are in reverse order. The slice localizer network is trained to output the probability that the image represent a slice at the top of the heart or the bottom. For

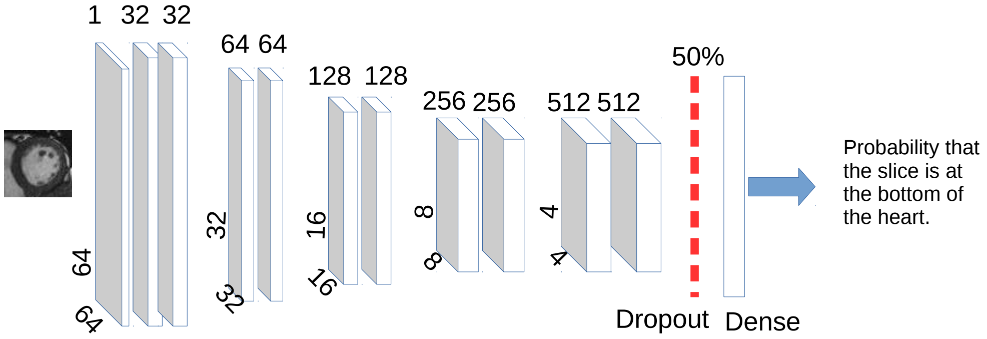

Figure 4.4: Slice Localizer Network: this network outputs a probability that a certain image in a slice is located at the bottom (or base) of the heart. In other words, if the image is of a slice at the base of the heart, the network will output a high probability. If the image is showing a slice near the top of the heart, the network will output a low probability.

instance, in subject 12, the slice localizer network states that the image in the first slice has a 98.2% probability of being in the bottom of the heart. In addition, the last slice has a 15.1% probability of being a slice in the bottom of the heart. Therefore, we should reverse the slices to make them consistent. This process ensures that for all subjects, the slices are ordered. This is important because the volume estimator network (Section 4.4) accepts each slice as a separate channel and expects the slices to be ordered and consistent.

4.4 Volume Estimation

In Section 4.2, we localized the left ventricle in the image. After that (in Section 4.3), the images were re-sized using the pixel spacing meta field in the DICOM files to ensure that the left ventricle cavity area is consistent across all images. Furthermore, a slice localization network was used to predict the slice location. The slices were ordered so that they are consistent before passing them to the volume estimation network.

The end systole and end diastole for each slice are defined as the images with the smallest and largest blood pool, respectively. Therefore, for each slice, we threshold the images and the end systolic and end diastolic images are chosen to be the ones with the smallest and largest

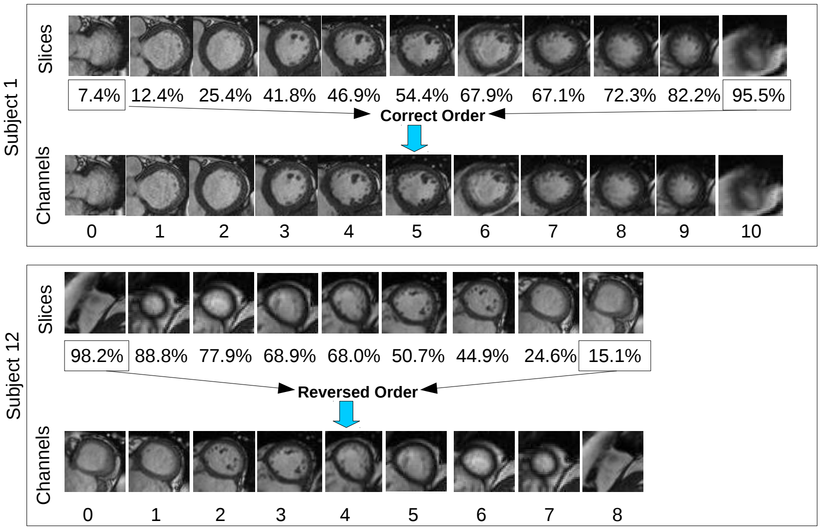

Figure 4.5: Ordering the Slices: this figure shows the end-diastolic image from each slice in subjects 1 and 12. The slices of subject 1 are in the correct order (from the top of the heart to the bottom). On the other hand, the slices of subject 12 are in reverse order. The slice localizer is trained to output the probability that the image is showing the top of the heart or the bottom. For instance, in subject 12, the slice localizer network states that the image in the first slice has a 98.2% of being in the bottom of the heart. In addition, the last slice has a 15.1% probability of being a slice in the bottom of the heart. Therefore, we should reverse the slices to make them consistent. This process ensures that for all subjects, the slices show the heart from top to bottom.

white pixels in one slice, respectively. The end systolic and end diastolic images from each slice form the channels of the input we pass to the volume estimation architecture.

As shown in Figure 4.6 , the input to the network is a set of images with size 96×96 pixels. The input has 36 channels (18 channels are for the diastole images and 18 channels for the systole images). In other words, we extract 1 systole image and 1 diastole image from each slice (a slice is a set of 30 images representing one heartbeat). We assume that the maximum number of slices possible is 18. After that, the input is passed through multiple sets of convolutional layers and maxpooling (8 convolutional layers & 1 maxpooling; 8 convolutional layers & 1 maxpooling; 4 convolutional layers & 1 maxpooling; 4 convolutional layers & 1 maxpooling; 2 convolutional layers & 1 maxpooling). The convolutional kernel size is (5,5). The activation for all convolutional layer is leaky rectification (with leakiness=0.1) [58, 26]. The network is regularized with dropout. The output layer contains 2 output units: one unit returns the

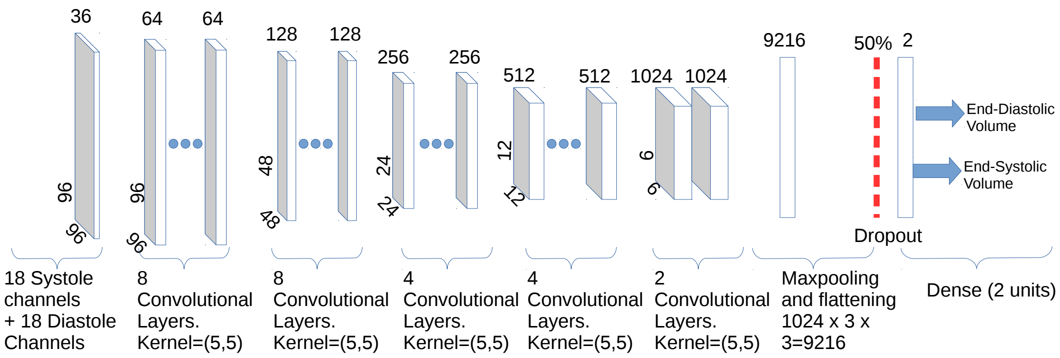
36
64
64
128
128
256
256
512
512
1024
1024
9216
50%
2
96
96
96
96
48
48
24
24
12
12
6
6
End-Diastolic Volume
End-Systolic Volume
Dropout
18 Systole channels + 18 Diastole Channels
8 Convolutional Layers. Kernel=(5,5)
8 Convolutional Layers. Kernel=(5,5)
4 Convolutional Layers. Kernel=(5,5)
4 Convolutional Layers. Kernel=(5,5)
2 Convolutional Layers. Kernel=(5,5)
Maxpooling and flattening 1024 x 3 x 3=9216
Dense (2 units)

Figure 4.6: Volume Estimation Architecture: the input to the network is a set of images with size 96×96 pixels. The input has 36 channels (18 channels are for the diastole images and 18 channels for the systole images). In other words, we extract 1 systole image and 1 diastole image from each slice (a slice is a set of 30 images representing one heartbeat). We assume that the maximum number of slices possible is 18. After that, the input is passed through multiple sets of convolutional layers and maxpooling (8 convolutional layers & 1 maxpooling; 8 convolutional layers & 1 maxpooling; 4 convolutional layers & 1 maxpooling; 4 convolutional layers & 1 maxpooling; 2 convolutional layers & 1 maxpooling). The convolutional kernel size is (5,5). The network is regularized with dropout. The output layer contains 2 output units: one unit returns the predicted end diastole volume and the other returns the predicted end systole volume.

predicted end diastole volume and the other returns the predicted end systole volume. We optimize the root mean square error between the true volumes and the predicted volumes.

In addition to regularizing the network with dropout, we augment the data by performing random transformations. These transformations include randomly rotating the input between -20 and 20 degrees. We also perform random horizontal and vertical flipping. Finally, we perform random Gaussian blurring (up to $\sigma = 1.0$).

4.5 Results

The model was trained on a laptop with the graphics card GTX980M (4GB dedicated RAM) with only 3.5GB available for training. Figure 4.7 shows the training and validation loss while training. The gap between the training loss and the validation loss remains almost the same throughout training. After 100 epochs, the validation error drops from 40 ml to around 15 ml.

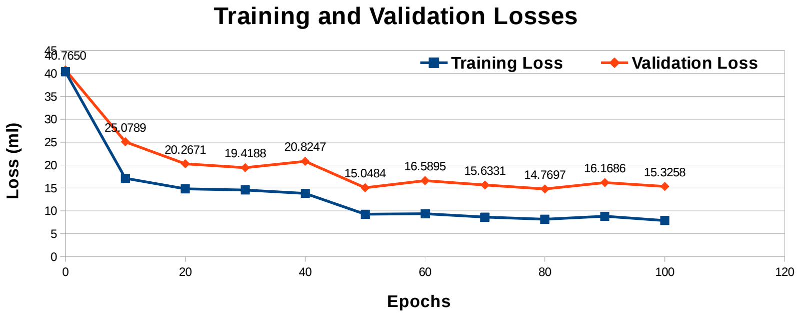

Figure 4.7: Training and Validation Loss: the figure shows how the training and validation loss (error) improves over the course of training. The loss (or the error) is measured in milliliter (ml).

Each epoch takes around 4 minutes to complete (total training time is 6.7 hours). On the other hand, predicting the volumes is done in real time.

The ejection fraction is probably the most important quantity since it measures the fraction of outbound blood pumped from the heart with each heartbeat. In general, low ejection fraction is an indication of heart problems. This quantity can be calculated as shown in Equation 4.4

$$EF = 100 \times \frac{(V_D - V_S)}{V_D} \tag{4.4}$$

where V_D and V_S are the end diastolic and end systolic volumes, respectively.

Table 4.1 shows a summary of the volume errors along with the ejection fraction. The mean absolute error (MAE) for the diastolic in milliliter (15.82 ml) appears to be higher than the mean absolute error for the systolic volume (11.16 ml). According to [60], these values are comparable to the performance of humans in estimating the volumes. The differences between two humans in estimating the end diastolic, end systolic, and ejection fraction are : 13 ml, 14 ml, and 6%, respectively. However, when looking at the root mean squared error (RMSE), there appears to be room for improvement. We believe that the RMSE values are higher than the MAE because there may be some outliers in the predicted volumes.

Table 4.1: Errors Summary - the table displays a summary of the volume errors in milliliter (for volumes) and percentage (for ejection fraction). Abbreviations: $MAE_D(ml)$ is the mean absolute error for the diastolic volume, $MAE_S(ml)$ is the mean absolute error for the systolic volume, $MAE_{EF}(\%)$ is the mean absolute error for the ejection fraction. $RMSE_D(ml)$ is the root mean squared error for the diastolic volume, $RMSE_S(ml)$ is the root mean squared for the systolic volume, $RMSE_{EF}(\%)$ is the root mean squared error for the ejection fraction.

$MAE_D(ml)$	$MAE_S(ml)$	$MAE_{EF}(\%)$	$RMSE_D(ml)$	$RMSE_S(ml)$	$RMSE_{EF}(\%)$
15.82	11.16	5.64	20.82	16.21	6.32

4.6 Conclusion

We described a model based on deep learning that can estimate the volume of the left ventricle. First, the left ventricle is localized. Then, the slice location is predicted and the slices of the study are ordered and preprocessed. After that, we identify the end systolic and end diastolic images for each slice. The end systolic and end diastolic images for each slice are stacked together and passed to a network that can estimate the end diastolic and end systolic volumes.

In the next Chapter, we will take a different approach to solve this problem. We will use the wide localization network in order to localize the left ventricle. Then, we will use it to segment the cavities. The standard localization network cannot perform segmentation. However, by using the wide localization network, we will be able to do that. It is worth mentioning that in the next chapter we will not use the whole dataset. Instead, we will use only 25 training subjects to training our model. Yet, the results we achieve is on par with the state of the art. This is mainly due to the localization operation that we perform which will simplify the problem significantly.

Chapter 5

Heart Volume and Ejection Fraction Prediction Using Wide Net

5.1 Introduction

Manually estimating ejection fraction and end-systolic and end-diastolic volumes from cardiac MRI images is a time consuming process. A cardiologist or an expert has to manually segment all slices before the volume can be calculated. We propose a deep and wide convolutional neural network that can be used to localize the left ventricle from an MRI image. Then, the systole and diastole images can be determined based on the size of the localized left ventricle. After that, the same network can be used to segment the cavity in the left ventricle. Using the DICOM meta fields, we can compute the volume size.

We use a dataset of MRI images provided by the National Heart, Lung, and Blood Institute (NHLBI) [41]. The dataset contains 500 training studies, 200 validation studies, and 440 testing studies. This dataset includes both the MRI images along with the diastole and systole volumes. In order to train the localization and segmentation models, we annotated some

---

[0] The majority of this chapter was originally published in: AbdulWahab Kabani, and Mahmoud R. El-Sakka "Ejection Fraction Estimation Using a Wide Convolutional Neural Network", International Conference on Image Analysis and Recognition (ICIAR), Montreal, Canada, July 2017 [pre-print]. [39]

images. For the localization model, we annotated 1 image per study (out of 161,910 images) which accounts for around (0.3%) of the training images. For the segmentation model, we annotated only 25 training studies out of 500 training studies (5% of the number of studies) and 5 validation studies in order to monitor the segmentation model training.

Left ventricle volume estimation has been an active area of research [55, 57, 20, 44]. Recently CNNs were used in order to tackle this problem [52, 10, 37]. For instance, in [10] a U-Net [69] network was used in order to segment the left ventricle.

A Convolutional Neural Network (CNN) is a neural network that contains some layers with restricted connectivity. CNNs were introduced in [18] and achieved good results on the MNIST data set [51]. The success of CNNs on the MNIST [51] was reproduced on image classification problems. CNNs can now produce state of the art performance in many classification tasks and on challenging datasets such as [11, 73]. The success of CNNs is due to many reasons including large training data sets [11, 73] , powerful hardware, regularization techniques such as Dropout [33, 77], initialization methods [23], ReLU activations [61, 30], and data augmentation. Since 2012, many networks that can perform image recognition were introduced [48, 79, 75]. Recently, CNNs have been used to perform detection and localization [69, 74, 80, 14, 76].

CNNs typically require large training data sets in order to be successful. However, cleaning and standardizing the data can help a lot in alleviating this requirement. In this paper, we divide the task into several steps in order to ensure that the training process is successful. First, the left ventricle is localized (Section 5.2). Then, the systole and diastole images for each slice are determined. After that, the left ventricle segmentation process is described in Section 5.3. Finally, the volume of the left ventricle is estimated (Section 5.4). In Section 5.5, we present the results. We conclude our work in Section 5.6. An overview of our proposed solution is shown in Figure 5.1.

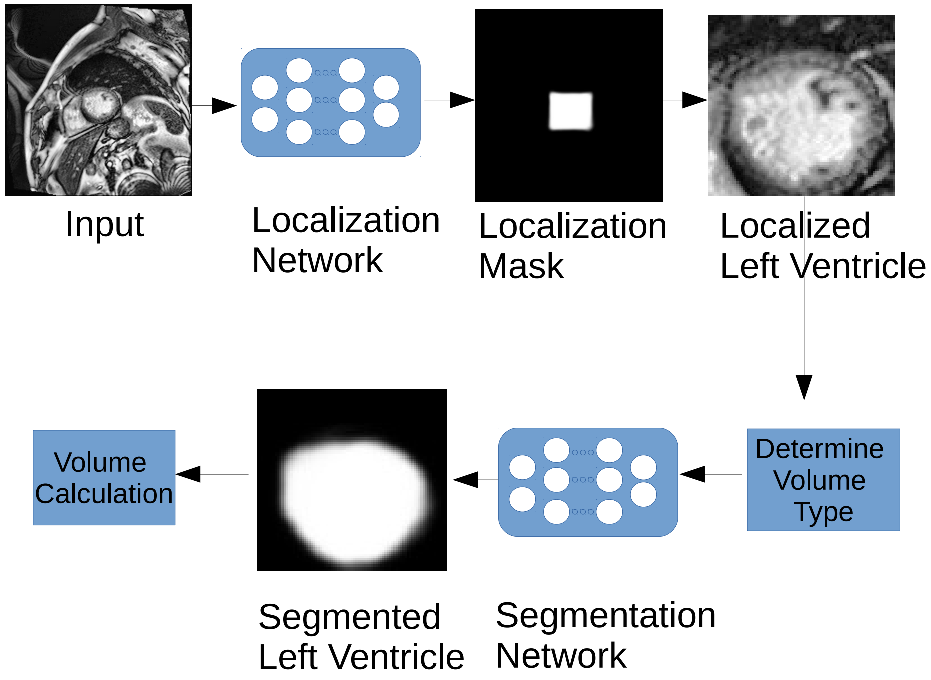

Figure 5.1: Overview: A set of training images (1 per subject) are used to train the localization network. Then, the network is used to predict a localization mask. The localization mask is thresholded using Otsu [65]. Once the left ventricle is localized, we can determine the systole and diastole images. The left ventricle is segmented by training a network on only 25 training subjects. After thresholding, the area of the left ventricle corresponds to the sum of pixel values that are not 0. The volume can be calculated by multiplying each area by the slice height and summing up the volumes.

5.2 Localization

The dataset contains (500 training subjects, 200 validation subjects, and 440 testing subjects). Each subject includes 8-16 short axis views (or slices). These slices provide a view of the heart at different levels. Each slice contains a set of (30 images or less) spanning one cardiac cycle. We are interested in only two images in each slice (the end-systolic and end-diastolic images). In each slice, these two images show us when the left ventricle is fully contracted and expanded, respectively. In order to determine these two phases, we first need to localize the left ventricle.

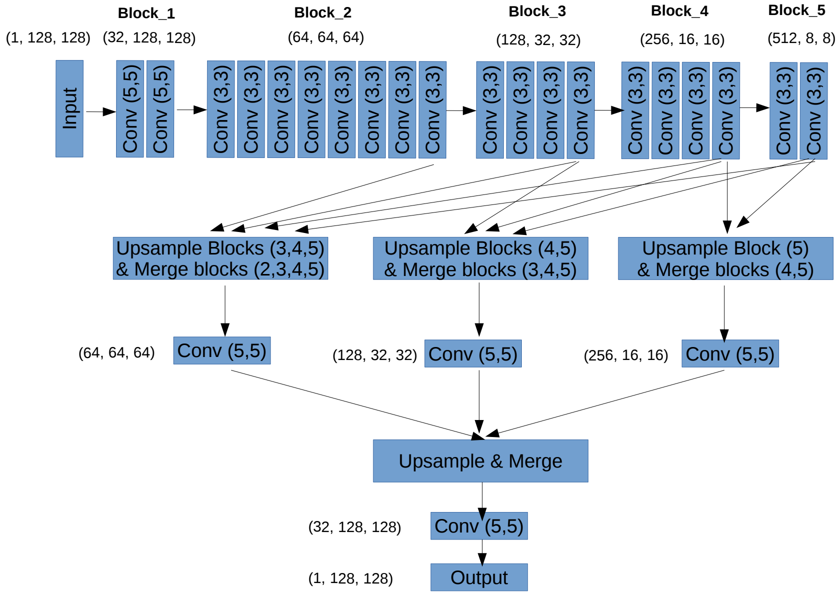

Figure 5.2: Network Architecture: This figure shows the architecture of one of the networks that were used for localization and segmentation of the left ventricle. The input goes through a set of convolutional layers. Then, the last layer from each block (except for block_1) are upsampled and merged in a hierarchical fashion in order to get the output mask in the original resolution.

We trained a localization network on a set of randomly sampled images from the training set. We made sure that one image is sampled from each patient. The architecture of the localization network is described in Figure 5.2. As shown in Figure 5.2, the input image is resized to (128, 128). Then, it is passed to the network and it goes through a set of convolutional layers. After that, the last layers from each block -except for the first one- are upsampled and merged. Three types of merging is done: type_1 involves merging (blocks_2, blocks_3, blocks_4, blocks_5), type_2 involves merging (blocks_3, blocks_4, blocks_5), and finally type_3 (merging block_4, and block_5). It is worth noting that type_1 carries more high resolution information while type_3 carries highly abstract information but with less resolution (due to downsampling).

Each convolutional layer is followed by rectified linear units (ReLU) activation [61, 30]. The output layer has a sigmoid activation to ensure that each pixel in the output mask ranges

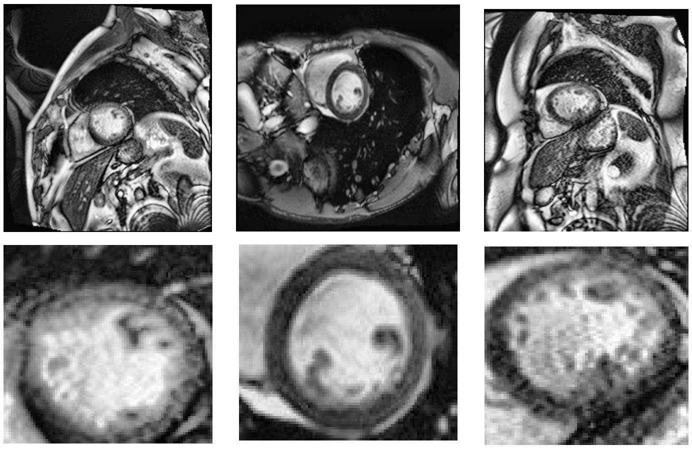

Figure 5.3: Localization Sample: First row shows the original images. Second row shows the localized left ventricle.

between 0 (black) to 1 (white). Each pre-activation is followed by batch normalization [36]. Maxpooling is used in order to extract more abstract features. We opted for upsampling by repeating the units rather than through parametrized upsampling in order to reduce the training time and GPU RAM consumption. Indeed, this network was also tested on a laptop with around 3.5 GB available GPU RAM.

The only input pre-processing that we did was subtracting the mean input of each channel and then dividing by the standard deviation. It is worth mentioning that in order for the left ventricle area to be determined correctly, we resized all images using the pixel spacing DICOM field. This ensures that each pixel represents 1mm. However, when training the localization network, we resized the input image to (128, 128) because the network can only accept fixed sizes. Once the network is used to predict the mask, the mask is resized to the original size where $1pixel = 1mm$. The following equations show how the pixel spacing is used to resize the image:

$$w_{new} = w_{old} \times \Delta s \tag{5.1}$$

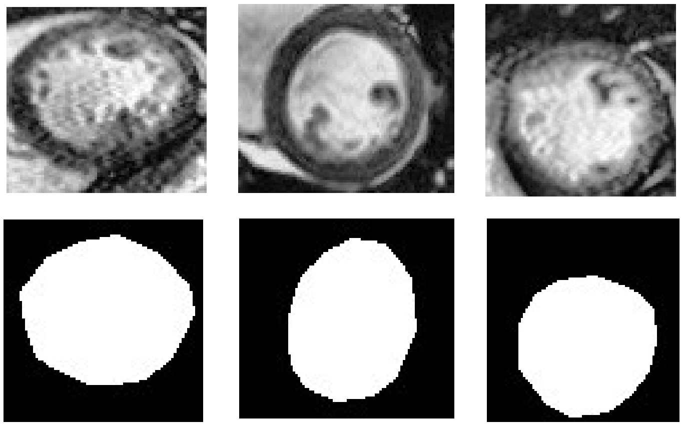

Figure 5.4: Segmentation Sample: First row shows the left ventricle images. Second row shows the segmentation mask.

$$h_{new} = h_{old} \times \Delta s \tag{5.2}$$

where w_{old} is the old image width, w_{new} is the new image width, h_{old} is the old image height, h_{new} is the new image height, and Δs is the pixel spacing, respectively.

The output of this step is shown in Figure 5.3.

5.3 Determining Interest Images and Segmentation

Because the size of the left ventricle varies between the end-systolic and end-diastolic phases, we can determine these phases by the size of the white area in the masks. For each slice, we consider the image with the smallest mask as end-systolic and the one with the largest mask as end-diastolic. A neural network with the same architecture as shown in Figure 5.2 can be used to segment the left ventricle. The network is trained only on the end-sysolic and end-diastolic images because these are the images that are needed to compute the end-systolic and end-diastolic volumes.

It should be noted that while the network has similar architecture to the localization network, the input size for the segmentation network is smaller (80×80) vs (128×128) for the localization network. This is because the input images are much smaller than the original images. Consequently, the segmentation network is trained much faster than the localization network. The segmentation network was trained only on 540 images representing 25 subjects. The training set contains 500 training subjects. We could not use the whole training set because the process of performing manual segmentation annotation is very time consuming. Because of the localization process, training the segmentation network on such a small data is possible as most of the irrelevant pixels are removed before segmentation.

Figure 5.4 shows a sample of left ventricle images and the predicted segmentation masks. These masks were generated by training an ensemble of 10 segmentation networks. The ensemble contains networks similar to the one described in Figure 5.2. We found that training an ensemble helps a lot since the number of training images is very low.

5.4 Volume Estimation

At this stage, we can calculate the volume because the left ventricle is segmented. We calculated the volume using the following equation:

$$V = \sum_{i=0}^{N} SA_i \times SH_i \tag{5.3}$$

where SA_i and SH_i are the area and height of the slice i, respectively. The area of the slice is calculated by taking the sum of all active pixels (nonzero pixels) in the segmentation mask. SH_i is the result of taking the absolute difference of slice locations from two consecutive slices.

Equation 5.4 shows how to compute the ejection fraction which is a measure of the outbound blood pumped from the heart after each heartbeat.

$$EF = 100 \times \frac{(V_D - V_S)}{V_D} \tag{5.4}$$

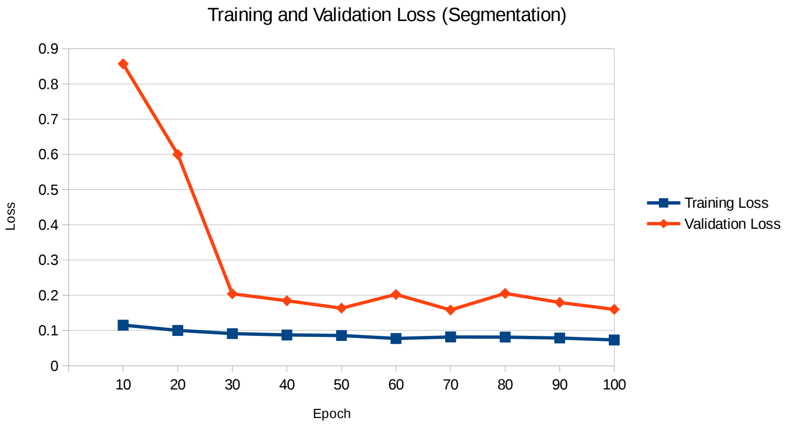

Figure 5.5: Segmentation Loss: the training and validation loss while training one of the segmentation models. There's a relatively big gap between the training and validation loss. This indicates that the model is likely overfitting the data.

where V_D and V_S are the end diastolic and end systolic volumes, respectively.

Because we did not train on the whole training set, we train four linear regression and random forest models to improve the results of the volume calculation. The features that are used to train these models include the systole and diastole volumes along with other features such as average slice height, average slice area, etc. The main reason for training these classifiers is to incorporate features other than the left ventricle images. For instances, age and sex can influence the volume of the heart. By training these classifiers, we are able to combine the predictions made by the neural network with the features stored in the DICOM file.

5.5 Results

Training the localization network for 500 epochs with learning rate 0.001 can take around 4.1 hours. On the other hand, training the segmentation network for 100 epochs takes only 28 minutes. Since we trained an ensemble of 10 segmentation networks, the segmentation training process takes around 5 hours. The mean absolute value error (MAE) we were able to achieve for the end-diastolic, end-systolic, and ejection fraction are 9.94 ml, 8.42 ml, and 4.47%, respectively. On the other hand, the RMSE errors are slightly higher (13.37 ml for the diastolic volume, 12.1 for the systolic volume and 5.97% for the ejection fraction). According to [60], these values are comparable to the performance of humans in estimating the volumes. The differences between two humans in estimating the end diastolic, end systolic, and ejection fraction are : 13 ml, 14 ml, and 6%, respectively. Tables 5.1, 5.2, and 5.3 shows a comparison between the results we achieved and the ones achieved by other methods.

In [52], they trained an ensemble of 10 models to predict the volumes. As part of the ensemble, some models can be used to segment the short axis images while other models can be used to segment the 4 chamber images. On the other hand, in [10], U-NET [69] was used to segment the left ventricle images. In [47], an ensemble of 250 models was developed. Some of the models are trained on the short axis images while the others were trained on the 2 chamber and 4 chamber images.

These results are very good considering the fact that the localization and segmentation CNNs were trained only on a small subset of the training. It is worth mentioning that many of the subjects in this dataset contained inconsistent data. For instance, some subjects contain MRI images that are repeated twice. This greatly affects the quality of the prediction. However, there are actions that can be done to improve the results. For instance, the results can further be improved by annotating more images. Figure 5.5 shows the training and validation losses while training one of the segmentation models. Because of the gap between the two losses, it is very likely that the model is overfitting the data. This issue is likely because we only annotated 25 subjects and used it to train the model. The results are likely to be improved if we annotate more subjects and train the model again.

Table 5.1: Diastole RMSE Comparison with other state of the art solutions. The state of the art solutions were obtained from [60]

Method	Diastolic RMSE (ml)
Tencia Woshialex [52]	12.02
Ours	**13.37**
JuliandeWit [10]	13.63
kunsthart [47]	13.65
ShowMeTheMoney	13.2

Table 5.2: Systole RMSE Comparison with other state of the art solutions. The state of the art solutions were obtained from [60]

Method	Systolic RMSE (ml)
ShowMeTheMoney	9.31
Tencia Woshialex [52]	10.19
JuliandeWit [10]	10.32
kunsthart [47]	10.43
Ours	**12.1**

5.6 Conclusion

We introduced a network that can be used to localize a region of interest in cardiac MRI images. Once the region of interest (left ventricle) is localized, the systole and diastole images are determined. After that, we segmented the left ventricle using the CNN we used for localization. Finally, we performed volume and ejection fraction estimation using the DICOM fields.

The main advantage of our model is that it was trained on only 25 subjects out of 500 subjects. Yet, the results are on par with the state of the art. This is mainly because we used the wide localization network to localize the left ventricle before performing the segmentation.

Table 5.3: Ejection Fraction RMSE Comparison with other state of the art solutions. The state of the art solutions were obtained from [60]

Method	Ejection Fraction RMSE (%)
ShowMeTheMoney	4.69
Tencia Woshialex [52]	4.88
JuliandeWit [10]	5.04
Ours	**5.97**
kunsthart [47]	6.99

Chapter 6

Conclusion

Deep learning is achieving state-of-the-art results on many problems with large datasets. Datasets such as medical images, astronomical images, and environmental images have very small training datasets. Because of this, the use of deep learning on these solutions can be very challenging. We introduced two region of interest localization networks that can be used to simplify problems with a low number of training data. The standard localization network was discussed in chapters 2 and 4. While being able to localize the region of interest, this architecture suffers from the following problems:

- Since the last layer is dense (fully connected), the number of parameters can grow very quickly.
- The network cannot be trained to learn more than one ROI. For instance, when applying this network to the whale recognition problem, we had to train two networks: one was used to localize the bonnet and the other was used to localize the blow hole.
- The network cannot learn to segment an arbitrary shaped ROI. While the top layers has highly abstract information, the spatial resolution is lost due to the pooling layers.
- The output mask requires thresholding in order to produce a binary mask. Hence, the results may vary depending on the thresholding model.

To address these shortcomings, we introduced the wide localization network. This network was discussed in Chapters 3 and 5. This network has several advantages, such as:

- It is a very efficient architecture with fewer number of parameters.
- The network may have several output channels. This means that it can be trained to localize several ROIs simultaneously.
- The network can learn to identify an arbitrary shaped ROI including segmentation.
- The output mask can be thresholded simply by using the threshold value of 50%. This is because the last layer has a sigmoid activation.

Given these advantages and improvements, we recommend using the wide localization network instead of the standard localization network.

In Chapters 2 and 3, we discussed the Right whale recognition problem. We noticed that using a localization network enabled us to train a deep neural network to recognize the whales. Without the localization networks, we would not have achieve good results. In Chapter 2, we presented a solution that ranked in the top 5 best solutions on the NOAA dataset. In Chapter 3, we used the wide localization network to improve the results even further.

The left ventricle volume and ejection fraction estimation was discussed in Chapters 4 and 5. In Chapter 4, we discussed how the left ventricular volume can be estimated by stacking the images and then using a convolutional network to estimate the volume. In Chapter 5, we improved the results by training a wide net to localize the left ventricle. Following this, the left ventricle was segmented and the slices were integrated for volume estimation. In this approach, we noticed that localizing the left ventricle before segmentation helped in producing results that are on par with the state-of-the-art, while only training the network on part of the training set. We ended up using only 25 training subjects.

This work introduced localization networks as a means of improving image recognition. As a future work, it would be interesting to evaluate the wide localization network performance against some of the best semantic segmentation networks such as [56, 62, 17, 88, 29, 7, 90]. It is important to pre-train the wide localization network on the Imagenet dataset [11, 73] before making this comparison.

Bibliography

[1] Kevork N Abazajian, Jennifer K Adelman-McCarthy, Marcel A Agüeros, Sahar S Allam, Carlos Allende Prieto, Deokkeun An, Kurt SJ Anderson, Scott F Anderson, James Annis, Neta A Bahcall, et al. The seventh data release of the sloan digital sky survey. *The Astrophysical Journal Supplement Series*, 182(2):543, 2009.

[2] Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[3] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[4] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[5] Robert Bogucki. Which whale is it, anyway? face recognition for right whales using deep learning. `http://deepsense.io/deep-learning-right-whale-recognition-kaggle/`. [Online; accessed 04-August-2016].

[6] G. Bradski. Opencv. *Dr. Dobb's Journal of Software Tools*, 2000.

[7] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[8] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3640–3649, 2016.

[9] Franois Chollet. keras. `https://github.com/fchollet/keras`, 2015.

[10] Julian de Wit. Third place solution for the second kaggle national datascience bowl. `https://github.com/juliandewit/kaggle_ndsb2`. [Online; accessed 01-February-2017].

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[12] Sander Dieleman, Kyle W Willett, and Joni Dambre. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly Notices of the Royal Astronomical Society*, 450(2):1441–1459, 2015.

[13] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[14] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2154, 2014.

[15] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[16] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[17] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013.

[18] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[19] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[20] Guido Germano, Hosen Kiat, Paul B Kavanagh, Mady Moriel, Marco Mazzanti, Hsiao-Te Su, Kenneth F Van Train, and Daniel S Berman. Automatic quantification of ejection fraction from gated myocardial perfusion spect. *Journal of Nuclear Medicine*, 36(11):2138, 1995.

[21] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[22] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):142–158, 2016.

[23] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256, 2010.

[24] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[25] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[26] Benjamin Graham. Spatially-sparse convolutional neural networks. *arXiv preprint arXiv:1409.6070*, 2014.

[27] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

[28] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 447–456, 2015.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, pages 346–361. Springer, 2014.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[32] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[33] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[34] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[35] Matthias Holschneider, Richard Kronland-Martinet, Jean Morlet, and Ph Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets*, pages 286–297. Springer, 1990.

[36] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[37] AbdulWahab Kabani and Mahmoud R. El-Sakka. Estimating ejection fraction and left ventricle volume using deep convolutional networks. In *International Conference Image Analysis and Recognition*, pages 678–686. Springer, 2016.

[38] AbdulWahab Kabani and Mahmoud R. El-Sakka. North atlantic right whale localization and recognition using very deep and leaky neural network. *Mathematics for Applications*, 2016.

[39] AbdulWahab Kabani and Mahmoud R. El-Sakka. Ejection fraction estimtion using a wide convolutional neural network. In *International Conference Image Analysis and Recognition*. Springer, 2017.

[40] AbdulWahab Kabani and Mahmoud R. El-Sakka. Improving right whale recognition by fine-tuning alignment and using wide localization network. In *Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference on*. IEEE, 2017.

[41] Kaggle. Data science bowl cardiac challenge data. https://www.kaggle.com/c/second-annual-data-science-bowl. [Online; accessed 19-March-2016].

[42] Kaggle. Diabetic retinopathy detection. `https://www.kaggle.com/c/diabetic-retinopathy-detection/`. [Online; accessed 19-January-2016].

[43] Kaggle. Right whale recognition. `https://www.kaggle.com/c/noaa-right-whale-recognition`. [Online; accessed 19-January-2016].

[44] Michael R Kaus, Jens von Berg, Jürgen Weese, Wiro Niessen, and Vladimir Pekar. Automated segmentation of the left ventricle in cardiac mri. *Medical image analysis*, 8(3):245–254, 2004.

[45] Christin Khan, Peter Duley, Allison Henry, Jennifer Gatzke2, and Timothy Cole1. North atlantic right whale sighting survey (narwss) and right whale sighting advisory system (rwsas) 2013 results summary. *US Dept Commer, Northeast Fisheries Science Center Reference Document*, pages 14–11, 2014.

[46] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[47] Ira Korshunova. Diagnosing heart diseases with deep neural networks. `http://irakorshunova.github.io/2016/03/15/heart.html`. [Online; accessed 01-February-2017].

[48] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[49] Felix Lau. Recognizing and localizing endangered right whales with extremely deep neural networks. `http://felixlaumon.github.io/2015/01/08/kaggle-right-whale.html`. [Online; accessed 04-August-2016].

[50] Quoc V Le. Building high-level features using large scale unsupervised learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8595–8598. IEEE, 2013.

[51] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[52] Tencia Lee and Qi Liu. Solution to win the second annual data science bowl. `https://github.com/woshialex/diagnose-heart`. [Online; accessed 01-February-2017].

[53] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks with identity mappings for high-resolution semantic segmentation. *arXiv preprint arXiv:1611.06612*, 2016.

[54] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[55] Xiang Lin, Brett R Cowan, and Alistair A Young. Automated detection of left ventricle in 4d mr images: experience from a large study. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2006*, pages 728–735. Springer, 2006.

[56] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[57] Michael Lynch, Ovidiu Ghita, and Paul F Whelan. Automatic segmentation of the left ventricle cavity and myocardium in mri data. *Computers in biology and medicine*, 36(4):389–407, 2006.

[58] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, volume 30, 2013.

[59] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[60] Jonathan Mulholland. Leading and winning team submissions analysis. `http://www.datasciencebowl.com/leading-and-winning-team-submissions-analysis/`. [Online; accessed 04-August-2016].

[61] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[62] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1520–1528, 2015.

[63] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

[64] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724, 2014.

[65] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27, 1975.

[66] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[67] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[68] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[69] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015.

[70] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[71] Frank Rosenblatt. *Principles of neurodynamics*. Spartan Book, 1962.

[72] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[73] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[74] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[75] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[76] Hyun Oh Song, Ross Girshick, Stefanie Jegelka, Julien Mairal, Zaid Harchaoui, and Trevor Darrell. On learning to localize objects with minimal supervision. *arXiv preprint arXiv:1403.1024*, 2014.

[77] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[78] Nervana Systems. Neon. https://github.com/NervanaSystems/neon, 2016. [Online; accessed 04-August-2016].

[79] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[80] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. In *Advances in Neural Information Processing Systems*, pages 2553–2561, 2013.

[81] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[82] Anil Thomas. whale-2015. https://github.com/anlthms/whale-2015, 2015. [Online; accessed 19-January-2016].

[83] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014.

[84] Laurie Von Melchner, Sarah L Pallas, and Mriganka Sur. Visual behaviour mediated by retinal projections directed to the auditory pathway. *Nature*, 404(6780):871–876, 2000.

[85] Kyle W Willett, Chris J Lintott, Steven P Bamford, Karen L Masters, Brooke D Simmons, Kevin RV Casteels, Edward M Edmondson, Lucy F Fortson, Sugata Kaviraj, William C Keel, et al. Galaxy zoo 2: detailed morphological classifications for 304 122 galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, page stt1458, 2013.

[86] Zifeng Wu, Chunhua Shen, and Anton van den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *arXiv preprint arXiv:1611.10080*, 2016.

[87] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[88] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[89] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[90] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. *arXiv preprint arXiv:1612.01105*, 2016.

[91] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537, 2015.

[92] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *arXiv preprint arXiv:1608.05442*, 2016.

Appendix A

List of Papers Published During My Doctoral Studies

A.1 Publications included in this thesis:

1. **Chapter 2**: AbdulWahab Kabani, and Mahmoud R. El-Sakka "North American Right Whale Recognition using Very deep and Leaky Neural Network", Journal of Mathematics Applications, Vol 5., 2016. [38]

2. **Chapter 3**: AbdulWahab Kabani, and Mahmoud R. El-Sakka "Improving Right Whale Recognition by Fine-tuning Alignment and Using Wide Localization Network", Canadian Conference on Electrical and Computer Engineering (CCECE), Windsor, Canada, May 2017 [pre-print]. [40]

3. **Chapter 4**: AbdulWahab Kabani, and Mahmoud R. El-Sakka "Estimating Ejection Fraction and Left Ventricle Volume using Deep Convolutional Networks", International Conference on Image Analysis and Recognition (ICIAR), Pvoa de Varzim, Portugal, July 2016. [37]

4. **Chapter 5**: AbdulWahab Kabani, and Mahmoud R. El-Sakka "Ejection Fraction Estima-

tion Using a Wide Convolutional Neural Network", International Conference on Image Analysis and Recognition (ICIAR), Montreal, Canada, July 2017 [pre-print]. [39]

A.2 Publications not included in this thesis:

1. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Object Detection and Localization using Deep Convolutional Networks with Softmax Activation and Multi-class Log Loss", International Conference on Image Analysis and Recognition (ICIAR), Pvoa de Varzim, Portugal, July 2016.

2. AbdulWahab Kabani, and Mahmoud R. El-Sakka "How Important is Scale in Galaxy Image Classification?", The International Conference on Computer Vision Theory and Applications (VISAPP 2016), Rome, Italy, February 2016.

3. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Adaptive Weighted Neighbors Lossless Image Coding", International Conference on Image Analysis and Recognition (ICIAR), Niagara Falls, Canada, June 2015.

4. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Weighted Ratio-Based Adaptive Lossless image coding", Canadian Conference on Electrical and Computer Engineering (CCECE), Toronto, Canada, May 2014.

Appendix B

Copyright Release

This appendix includes some information about the copyright in each of the chapters.

Chapter 2: the majority of this chapter was originally published in "Journal of Mathematics Applications" [38]. I am the main author for this paper. My supervisor is a co-author. We retain the copyright for this paper.

Chapter 3: the majority of this chapter was originally published in "Canadian Conference on Electrical and Computer Engineering" [40]. I am the main author for this paper. My supervisor is a co-author. According to the IEEE frequently asked questions, it is acceptable to use the paper in a dissertation or thesis as long as the following two statements are included:

- ©[2017] IEEE
- In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of University of Western Ontario's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to
  http://www.ieee.org/publications_standards/publications/rights/rights_link.html
  to learn how to obtain a License from RightsLink. If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the

dissertation.

Chapter 4: the majority of this chapter was originally published in “International Conference on Image Analysis and Recognition (ICIAR) 2016 ” [37]. I am the main author for this paper. My supervisor is the co-author. We own the copyright for this paper. Springer (Lecture Notes in Computer Science) is the publisher. According to the Springer Lecture Notes in Computer Science copyright policy form: “Author retains the right to use his/her Contribution for his/her further scientific career by including the final published paper in his/her dissertation or doctoral thesis provided acknowledgment is given to the original source of publication.”

Chapter 5: the majority of this chapter was originally published in “International Conference on Image Analysis and Recognition (ICIAR) 2016 ” [37]. I am the main author for this paper. My supervisor is the co-author. We own the copyright for this paper. Springer (Lecture Notes in Computer Science) is the publisher. According to the Springer Lecture Notes in Computer Science copyright policy form: “Author retains the right to use his/her Contribution for his/her further scientific career by including the final published paper in his/her dissertation or doctoral thesis provided acknowledgment is given to the original source of publication.”

Curriculum Vitae

Name:	AbdulWahab Kabani
Post-Secondary Education and Degrees:	The University of Western Ontario London, ON, Canada 2012-2017 Ph.D.
	American University of Sharjah Sharjah, UAE 2005 - 2009 BSc.
Honours and Awards:	Halyard Ultrasound Nerve Segmentation - First Prize Winner Value: USD $50,000 Institute: Halyard Health and Kaggle August 2016
	Ranked in top 28 (out of 48,631 data scientists) on Kaggle Institute: Kaggle August 2016
	Queen Elizabeth II Graduate Scholarship in Science and Technology Ontario Graduate Scholarship (OGS) Institute: The University of Western Ontario Value: $15,000 per year Years: May 2016
	Best Research Talk Award in Computer Vision and Image Processing Institute: University of Western Ontario Research in Computer Science (UWORCS) Years: 2013, 2014, 2015, 2016
	Chancellor's List Institute: American University of Sharjah Years: 2007

Dean's List
Institute: American University of Sharjah
Years: 2006-2009

Related Work Experience:

Technical Consultant
eSolutions FZ-LLC (IBM Premier Partner)
2010-2012

Teaching Assistant
The University of Western Ontario
2012-2016

Publications:

1. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Ejection Fraction Estimation Using a Wide Convolutional Neural Network", International Conference on Image Analysis and Recognition (ICIAR), Montreal, Canada, July 2017 [pre-print].

2. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Improving Right whale Recognition by Fine-tuning Alignment and Using Wide Localization Network", Canadian Conference on Electrical and Computer Engineering (CCECE), Windsor, Canada, May 2017 [pre-print].

3. AbdulWahab Kabani, and Mahmoud R. El-Sakka "North American Right whale Recognition using Very deep and Leaky Neural Network", Journal of Mathematics Applications, Vol 5., 2016.

4. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Estimating Ejection Fraction and Left Ventricle Volume using Deep Convolutional Networks", International Conference on Image Analysis and Recognition (ICIAR), Pvoa de Varzim, Portugal, July 2016.

5. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Object Detection and Localization using Deep Convolutional Networks with Softmax Activation and Multi-class Log Loss", International Conference on Image Analysis and Recognition (ICIAR), Pvoa de Varzim, Portugal, July 2016.

6. AbdulWahab Kabani, and Mahmoud R. El-Sakka "How Important is Scale in Galaxy Image Classification?", The International Conference on Computer Vision Theory and Applications (VISAPP 2016), Rome, Italy, February 2016.

7. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Adaptive Weighted Neighbors Lossless Image Coding", International Conference on Image Analysis and Recognition (ICIAR), Niagara Falls, Canada, June 2015.

8. AbdulWahab Kabani, and Mahmoud R. El-Sakka "Weighted Ratio-Based Adaptive Lossless image coding", Canadian Conference on Electrical and Computer Engineering (CCECE), Toronto, Canada, May 2014.
9. Tamer Shanableh, Khaled Assaleh, Layla Al-Hajjaj, and AbdulWahab Kabani "Gait Recognition System Tailored For Arab Costume of the Gulf Region", IEEE Symposium on Signal Processing and Information Technology (IEEE ISSPIT), Ajman, UAE, December 2009.