VIETNAM GENERAL CONFEDERATION OF LABOUR
TON DUC THANG UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY

_____ * _____

GRADUATION THESIS

AUTOMATED MEASUREMENT OF FETAL HEAD CIRCUMFERENCE USING 2D ULTRASOUND IMAGES

Instructing Lecturer: **PhD LE MINH HUNG**
Student's Name: **TRAN TUAN CANH**
Class: **17050311**
Course: **21**

HO CHI MINH CITY, YEAR 2021

VIETNAM GENERAL CONFEDERATION OF LABOUR
TON DUC THANG UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY

_____ * _____

GRADUATION THESIS

AUTOMATED MEASUREMENT OF FETAL HEAD CIRCUMFERENCE USING 2D ULTRASOUND IMAGES

Instructing Lecturer: **PhD LE MINH HUNG**
Student's Name: **TRAN TUAN CANH**
Class: **17050311**
Course: **21**

HO CHI MINH CITY, YEAR 2021

ACKNOWLEDGEMENT

During the past time, we could not have been done without the kind support from many individuals and organizations – which is why I would like to send my most profound gratitude to all of them.

First of all, I would like to express a special appreciation to Ton Duc Thang University's teachers for their conscientious guidance, we has learned a lot of useful things and accumulated some knowledge to complete into this report.

I am highly indebted to Mr. Le Minh Hung for his guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project. Again, I sincerely thank you.

I would like to express my special gratitude and thanks to industry persons for giving me such attention and time.

My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.

HO CHI MINH CITY, March 25, 2021

Author

Tran Tuan Canh

THE PROJECT WAS COMPLETED AT TON DUC THANG UNVERSITY

I pledge that is our report project and guided by Mr. Le Minh Hung. The content of research, results in this topic are honest and not published in any form before. The data in the tables used for the analysis, comment, and evaluation were collected by the authors themselves from various sources indicated in the reference section. In addition, in this exercise, a number of comments, reviews and other authors' data are available.

If any fraud is found, I am fully responsible for the content of my report. Ton Duc Thang University is not involved in any copyright infringement or copyright infringement in the course of implementation.

Ho Chi Minh city, March 25, 2021

Author

Tran Tuan Canh

EVALUATION OF INSTRUCTING LECTURER

Confirmation of the instructor

.....

.....

.....

.....

.....

Ho Chi Minh city, day month year

(Full name, Signed and Sealed)

The assessment of the teacher marked

.....

.....

.....

.....

.....

Ho Chi Minh city, day month year

(Full name, Signed and Sealed)

Contents

LIST OF ABBREVIATIONS

Medical terms:

- HC: Head Circumference
- CRL: Crow-Rump Length
- GA: Gestational Age

Deep learning terms:

- MLP: Multi Layer Perceptron
- SVM: Support Vector Machine
- CNN: Convolution Neural Network
- GD: Gradient Descent
- SGD: Stochastic Gradient Descent
- R-CNN: Region Based Convolutional Neural Network
- RPN: Region Proposal Network
- FPN: Feature Pyramid Network
- IoU: Intersection over Union
- FC layer: Fully Connected layer
- FCN: Fully Convolutional Network

List of Tables

List of Figures

Abstract

In this paper, we introduce a segmentation system that can automatically measure the fetal head circumference (HC) in real-time by using ultrasound images. With the optimized results, the HC can be applied to estimate the gestational age and keep track on the health of the infants. Therefore, it can help solving the shortage of well-trained specialists, doctors, and sonographers. The system includes three core stages that are responsible for specific task. The first stage is a detection and segmentation stage which adopts convolutional neural network to generate a mask for the fetal head. After that, we optimize the result of the mask by using an ellipse fitting algorithm that smoothing the edge of the mask. Finally, in the third stage, a simple ellipse perimeter calculation is applied to estimate the HC. The system was trained, and tested on 999 ultrasound images and 335 ultrasound images respectively. The model effectively shows a robust performance with the detection loss and segmentation loss are 0.0129 and 0.0656 respectively in validation stage. Thus, it outperforms many traditional systems that employ handcrafted methods to estimate the HC.

Chapter 1

Introduction

In the modern world of medical field, ultrasound images are the most broadly used to monitor and diagnosis fetus during pregnancy. Many women prefer to use it to follow the development of fetus because it is economical, real-time monitoring, and extremely safe compared to X-rays or others types of imaging systems that use radiation based-method [citation]. Ultrasound examinations provide parents with a valuable opportunity to understand the health status of their unborn child. While ultrasound is considered to efficient and safe, it still has several disadvantages. The process of ultrasound imaging depends mostly on the one who operates the system for its signature noises on image such as shadows and reverberations which makes the image extremely hard to observe and diagnose the fetus' status. That is why ultrasound system can only use by well-trained, professional specialists, doctors, and sonographers which leads to the shortage human resources in poor and developing countries.

In the fetal ultrasound image, ultrasound allow the visualization of some body features, possibly other parts such as fingers and toes of the fetus. Based on these important features, many biometric measurements are applied by doctors or ultrasound imaging operators. For example, the crow-rump length (CRL) and head circumference (HC) are commonly calculated to estimate the gestational age (GA) and given diagnosis about growth of the fetus. The CRL widely uses for its accuracy to estimate the GA of the fetus in the age between 8 weeks 4 days and 12 weeks. After 13 weeks, specialists usually use HC for its accuracy instead of CRL. The instruction of HC measurement state that HC should be measured in a transverse section of the head with a central midline echo, interrupted in the anterior third by the cavity of the septum pellucidum with the anterior

and posterior horns of the lateral ventricles in view [5]. The HC measurement steps are proceeded manually which make the result unstable when being conducted by different doctors. The idea of creating an automated system is born base on the above obstacle. With the support from computer, the HC measurement result will be no longer affected by observer variability along with the shortage of sonographers.

Chapter 2

Related Works

In the past decades, many systems for automatic HC measurement have been introduced with various traditional approaches. In 2005, Wei Lu et al presented a system that used a low-pass filter to reduce noise in ultrasound images and a transformation filter to increase contrast between skull and background (this process method is based on the signature of bones on ultrasound image that usually brighter than the background). And they optimized K-mean algorithm and morphologic binary area opening operation to achieved skull segment. After that, Wei Lu et al assumed the skull was elliptical shape and then applied an iterative randomized Hough transform to predict the offset value of the ellipse. Their result was quite remarkable with the differences between sonographers and the system only 0.52% while predicting HC [10].

In 2017, Jing li et al was inspired by ellipse fitting methods, and proposed a system that employed a random forest algorithm to locate the fetal head. Then, they used phrase symmetry algorithm to detect the ellipse center and applied a non-iterative ellipse fitting method to efficiently fit the ellipse on the fetal head. As a result, their method archived an average measurement error of 1.7 mm and outperformed traditional methods [8].

In the next year, Thomas L. A. van den Heuvel et al introduced a method included three systems that measure the HC in all trimester of pregnancy. Each system architecture has different number of pipelines that employs a random forest algorithm to extract Haar-like features and a set of Hough transform, dynamic program, ellipse fitting algorithm to measure the HC. Therefore, by focusing on the nature of each trimester, this system showed not only the feasibility but also the robustness on fetal heads of each trimester [5].

Many systems for automatic HC measurement have been introduced with various tra-

ditional approaches using Hough transform, Haar-like features, ellipse fitting, etc. giving promising results. However, variations on the dataset are high due to noise, format, screening parameters configurations, etc. Therefore, traditional methods which based on hand-crafted features are not robust to all the variations of the images.

Base on others research, we consider this problem is not only the detection task but it is, specifically, the segmentation task that separates the fetal head from the noisy background. In this research, we focus on a more novel method for fetal head detection and segmentation using convolutional neural networks (CNNs) that show the robustness on a lot of computer vision tasks [14], [6], [13].

Many models in this research have been utilized to evaluate the performance of different CNNs configuration to analyze the insight of the dataset. The CNN model we used was fine-tuned and trained on 999 ultrasound images, and then it was tested on 335 ultrasound images of the HC-18 dataset which is public on grand-challenge.org. The model significantly shows robustness on the dataset with the detection loss and segmentation loss are 0.0129 and 0.0656 respectively in validation stage.

Chapter 3

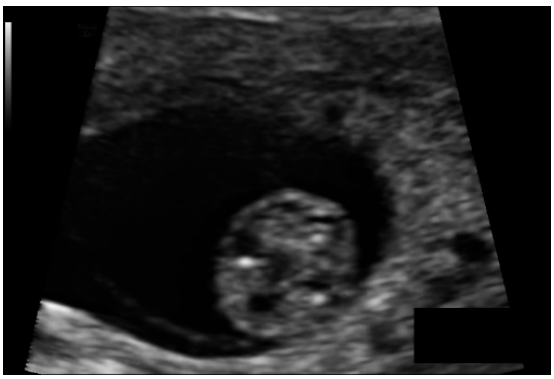
Materials And Methods

3.1 HC18 Dataset

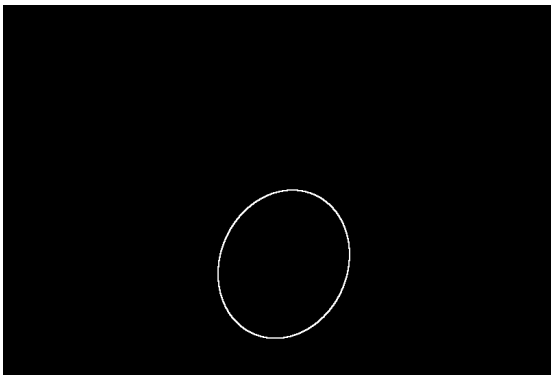
3.1.1 Description

The collection of 1334 2D ultrasound images of fetal head were collected from the database of the Department of Obstetrics of the Radboud University Medical Center, Nijmegen, the Netherlands. All of the fetal head ultrasound images for this challenge were captured in the standard plane which is the specifically used to measure the HC [5].

The data is divided into a training set of 999 images and a test set of 335 images. The size of each 2D ultrasound image is 800x540 pixel with a pixel size ranging from 0.052 to 0.326 mm. The information of pixel of each image can be found in the CSV files: 'training-set-pixel-size-and-HC.csv' and 'test-set-pixel-size.csv'. The variability of pixel size was resulted from the adjustment of sonographers during examinations which led to a different shape and size of the fetal heads. In addition, in the training set, come along with each ultrasound image is a manual annotation in millimeters. All of the image file-names start with a number. However, the file-names only set to 805 because some images were come from the same examination (different frame during monitoring), therefore they have a similar appearance [5].



(a) A fetal head ultrasound image.

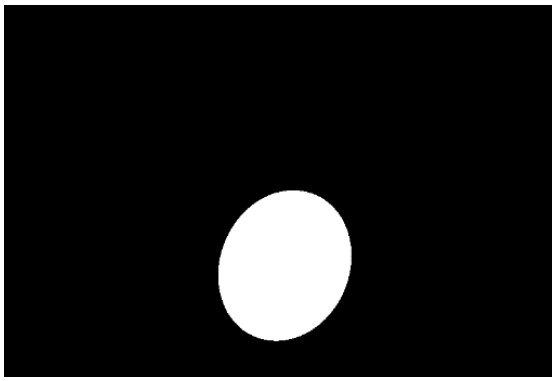


(b) An annotation created by sonographers.

Figure 3.1: HC-18 dataset sample.

3.1.2 Pre-processing

During each examination, the sonographer manually draws an ellipse which best fit the fetal head and save it as annotation of the ultrasound image. And based on these annotations, we apply a few image processing to extract the ellipse coordinates and created a dataset in COCO format for further training methods.



(a) a processed annotation

Figure 3.2: This annotation is made for Mask R-CNN model.

And then we save it in json file - "annotations.json" with the information structure including images, categories, and annotations.

Images:

- File name and ID.
- Height and width: size of each image.

Categories:

- Super-category: name for group of objects (for example: fish – which include lots of fish species).
- ID and name

Annotations:

- Segmentation: coordinates of annotated pixels (contour).
- Area: ground true binary mask.
- Is crowd: one or multiple objects in the image.
- Image ID, ID, category ID.
- Bounding box: offset values.

3.2 Convolutional neural network

3.2.1 A brief history of CNN

In the age of information and technology, the hardware resources have made some extraordinary breakthrough which enhances the parallel calculation power of both CPU and GPU. Based on these success, artificial intelligence field is back to the race after decades being hold back because of the limitation of hardware.

In the recent year, "Deep Learning" has become the most popular keywords in artificial intelligence field. However, its history was date back to the late 1950s with the invention

of perceptron architecture. Being influenced by the capability of biological system on perceptual recognition, generalization, recall, and thinking, F. Rosenblatt et al tried to mimic a brain functions by focusing his research on the smallest element of the brain - neurons. The author claimed that perceptron's design could simulate several signature features of the intelligence systems at an acceptable standard - a better-than-chance probability without applying multiple techniques to create a special environment of a specific biological creature. As a result, Rosen blatt believed that the fundamental laws of all physical system and humans could eventually be understood [12]. Even though perceptron showed a promising future for the intelligence system, the problem perceptron could not encounter was XOR. To be more specific, XOR operation is a linear un-division operation which cannot be represented by a single-layer perceptron. Hence, multi-layer perceptron (MLP) was adopted to solve the issue [15]. However, it was extremely hard and ineffective to train the MLP back then.

Therefore, not until the 1980s, MLP with backpropagation algorithm was finally introduced as an upgrade of the original perceptron that solve the image classification problems [1]. These algorithms achieved a few successes but then being constrained because computer at that time was not strong enough to run such heavy model (not mention the quality and quantity of the data).

After the birth of Support Vector Machine (SVM) algorithm which solved lots of disadvantages of perceptron models, deep learning once again being forgotten [cite SVM]. Not until 1998, Yann LeCun developed the model called LeNet, one of the first convolutional neural networks which has 2 layers (Convolution + Max pooling) and 2 fully connected layers and softmax layer as the output layer. Yann LeCun's model marked the comeback of deep learning after achieved significant accuracy (up to 99%) on MNIST dataset [7]. After the success of LeNet, lots of models had been developed and achieved promising results on image classification problem.

In 2012, another milestone was the winner of ImageNet LSVRC-2012 - AlexNet of Geoffey Hinton et al. This model was the breakthrough in deep learning field which opened

a new era for the revolution of neural network and contributed directly to a numerous artificial intelligence application recently. AlexNet is a huge model with 5 convolutional layer and 3 fully-connected layer (around 60 million parameters) was trained on ImageNet dataset which has around 1.2 million images of 1000 labels. The network achieved a top-5 error of 15.3, more than 10.8% points lower than its rivals. The original paper's primary result was that the depth of the model was essential for its high performance, which was computationally expensive, but made feasible due to the utilization of graphics processing units (GPUs) during training.

Since 2012, researchers around the world have started to create a numerous model from wider ones to deeper ones and achieved countless breakthrough and knowledge in deep learning field. Some outstanding models such as VGGNet (Karen Simonyan, Andrew Zisserman et al 2014), GoogleNet (Szegedy et al 2014), ResNet (Kaiming He et al 2015), etc. [cite VGG, GoogleNet, Resnet] have significant performance on computer vision tasks that allows computers work at human level.

3.2.2 A CNN architecture

In the previous section, we introduce briefly about CNN history, so what exactly is it? A typical CNN architecture consists of 3 types layers which are the input layer, the hidden layers, and the output layer. Unlike the traditional neural networks which only have layers of fully-connected neurons following by activation functions, CNNs also have layers that perform convolution, pooling layer to down size the feature map, and then the fully-connected layers.

Convolution layer

In the context of CNN, a convolution is a linear operation that involves the multiplication a set of weights with the input. The multiplication is performed between an input array and a weights array called filter or kernel. Let's say every image's pixel is a value in a 2D matrix. Convolution layer are the major building blocks used in neural network.

Each of convolution layer has a specific number of filters which slide over the image to extract its features by execute the dot product. A dot product is the element-wise multiplication between the filter-sized patch of the input and filter, which is then summed, always resulting in a single value. Because it results in a single value, the operation is often referred to as the "scalar product". [citation Dive into DL]

For example:

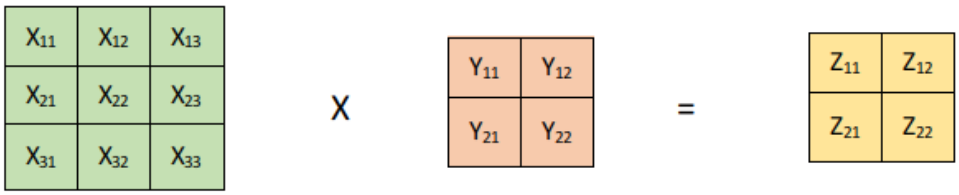


Figure 3.3: Example of convolution operation.

Calculation step as below:

$$Z_{11} = X_{11}Y_{11} + X_{12}Y_{12} + X_{21}Y_{21} + X_{22}Y_{22}$$

$$Z_{12} = X_{11}Y_{12} + X_{12}Y_{13} + X_{21}Y_{22} + X_{22}Y_{23}$$

$$Z_{21} = X_{11}Y_{21} + X_{12}Y_{22} + X_{21}Y_{31} + X_{22}Y_{32}$$

$$Z_{22} = X_{11}Y_{22} + X_{12}Y_{23} + X_{21}Y_{32} + X_{22}Y_{33}$$

This process seems simple, but it is very effective in image processing. In a CNN, the input is a tensor with shape (number of images) x (image height) x (image width) x (input channels). After passing through a convolution layer, the image becomes abstracted to a feature map (the result after applying convolution operation), with shape (number of images) x (feature map height) x (feature map width) x (feature map channels).

To train or modify a CNN, there are some attributes we should keep track on:

- The size and the number of filters/kernels (hyper-parameters).
- The number of input and output channels (hyper-parameters).

- The quantity of filters is equal to the quantity of feature maps.
- Padding and Stride attributes in convolution operations.

Padding and Stride are the crucial technique in convolution. Let's take a look at the previous example in Figure 1. We calculated the feature map by sliding the filter on the input data by one pixel for each convolution operation, we refer to the number of rows and columns traversed per slide as the stride. The smaller step we define in filters, the more information we can extract from it, but it is a trade-off which affect the training time. One tricky issue when applying convolution layers is that we tend to lose pixel information on the perimeter of input data.

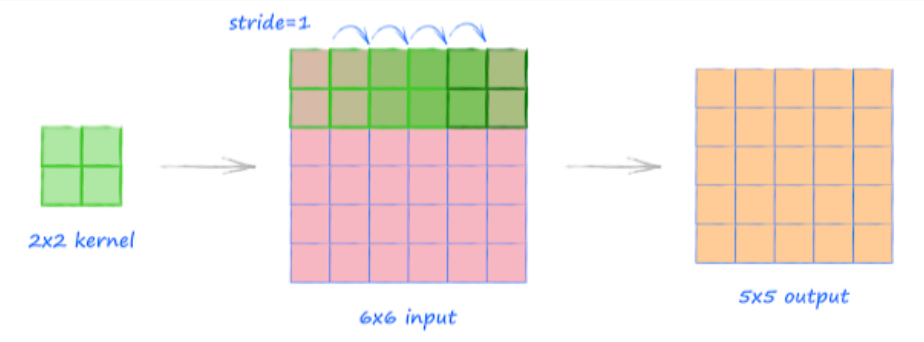


Figure 3.4: Missing pixels information when doing convolution.

This problem happens when we slide the filter on input data, our filters only do convolution operation on pixels at the edge of matrix fewer than pixels which are close to the center of the matrix. Even though, it is just a few pixels, we use lots of convolution layers, it will add up and cause missing information. That is why the term "Padding" comes up, a direct solution for this is to add a "barrier" of zero pixels surrounding the input matrix which increases the effective size of the input matrix.

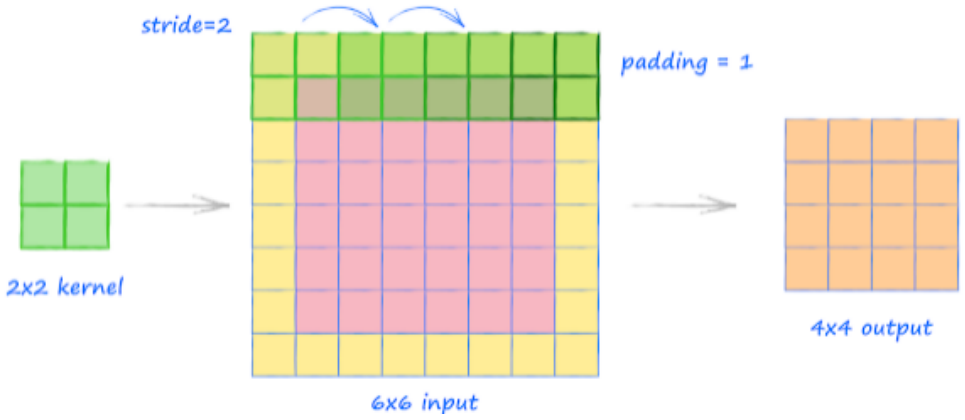


Figure 3.5: Applying zero padding = 1 to the input matrix.

Pooling layer

As described in previous section, the convolutional layer output is a feature maps, as it can be regarded as the learned representations (features) in the spatial dimensions (e.g., width and height) to the subsequent layer. Usually, as we process an image, we want to reduce the spatial resolution of our feature maps, select information so that it can be more robust or sensitive when we go deeper in the network. By gradually aggregating information, yielding coarser and coarser maps, our model can accomplish a global representation for the feeding dataset.

Like convolution, pooling operator consists of a fixed-shape window that slide all over on the output feature map according to its stride setting. However, unlike sliding window of convolutional layer, sliding window of pooling layer does not have any parameters (no kernel). Instead of calculating the feature map, pooling layer simply calculate either the maximum or average value of the pixels inside its sliding window. These operations are called maximum pooling and average pooling, respectively. Another characteristic of pooling layer is that it also can alter padding and stride settings to prevent model from missing information.

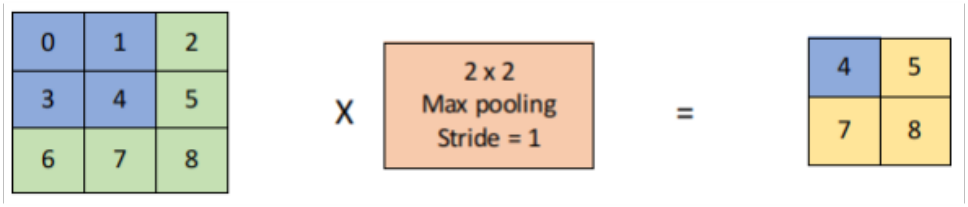


Figure 3.6: Max pooling with window shape 2x2 and stride = 1.

Fully connected layer

Fully-connected (FC) layers connect every neuron in one layer to every neuron in another layer. It is the similar to a traditional multi-layer perceptron neural network (MLP). After we extract feature maps from convolutional layers, we flatten these maps to one dimension vector and then feed it to the FC layer for the model to learn how to classify the data. However, FC layers can be seen as a brute force approach whereas there are approaches like the convolutional layer which reduces the input to concerned features only. That is the reason why CNN models are often built with few FC layers (one or two layers).

Loss function

In the training process, a CNN model learns to map a set of inputs to a set of outputs. By using gradient-based training method, the mapping process cannot be calculated perfectly in one step due to convergence process. Therefore, the model has to feed the inputs from the dataset over and over again to improve the mapping result. As a result, the problem of learning is explore as optimization problem.

Base on this idea, loss function (or cost function) is defined as the difference of the dataset output and the predicted output of the model. It is like a form of getting the model to pay a penalty after a wrong prediction, and loss function output is proportional to the severity of the error. In supervised learning problem, the goal is to minimize this loss as low as possible (In the ideal condition, loss function returns 0).

A general loss function is expressed as below:

$$L_\varepsilon\Big(y, f(x,w)\Big) = \begin{cases} 0 & \Big|y - f(x,w)\Big| \leq \varepsilon, \\ \Big|y - f(x,w)\Big| - \varepsilon & \text{otherwise,} \end{cases}$$

As loss function calculate the difference between the right and the wrong mapping value, naturally, it can be defined as a subtraction of the two values. However, subtracting 2 values may result in negative number which cannot be considered, therefore, an absolute expression is added to constrained its output.

$$L(\hat{y}\, y) = |\hat{y}\ - y|$$

Because the model is trained based on gradient based method, the minimization process of above function is hardly to achieve due to the un-continuous derivative (for example, derivative of $f(x) = |x|$ is interrupted at 0). To address this issue, the whole expression is squared and then divide by two.

A simple loss function:

$$L(\hat{y}\, y) = \frac{1}{2}\ (\hat{y}\ - y)^2$$

Let's take a binary classification problem to be an simple example which $\hat{y}\ < 0$ means the model prefers -1 prediction and $\hat{y}\ > 0$ means the model prefers 1 prediction. Hence, an appropriate loss function that satisfies the following criteria:

1. Whenever the model make a wrong predict, it must be punished badly. Therefore, the first criteria is that loss function has to return a bigger value if $\hat{y}$ is opposite to y than the contrast.

2. If there are two answers $\hat{y}$ and y both have the same sign (or different sign), which one should be punished more? As mentioned before, the absolute value $|\hat{y}\ |$ represents the "prefer-ness" of the model for an option. The larger this value, the more the model prefers an answer. In the case of the same sign, the preferred option is the correct one, so the more the model likes its prediction, the less penalties must be paid. With the same argument, if the sign $\hat{y}\ $ is different from y, because the preferred option is the wrong one,

the more the model likes, the more severe penalties will be imposed on the model so that the model will not repeat it.

Yet, with different problems, there are different loss functions with multiple criteria. Therefore, before applying any models, algorithms, researchers need to identify the problem so that they can choose a suitable loss function to train the model.

Optimization function

In the previous section, we discussed about the loss function and its criteria. Once we define a loss function to make penalty on the wrong prediction of the model, we need to minimize the loss value in attempt to make the model prediction better after each training step. Therefore, an algorithm called optimization algorithm was brought into the training process in order to serve the optimization problem in the training phrase.

Although optimization provides a way to minimize the loss function for deep learning, in essence, the goals of optimization and deep learning are fundamentally different. The former is primarily concerned with minimizing an objective whereas the latter is concerned with finding a suitable model, given a finite amount of data.

Despite the fact that optimization algorithm provides a method for deep learning model to minimize the loss function value, in general, the purpose of them is essentially different from one another. In deep learning field, we mostly concern find a model that can generalize a finite amount of given data. However, the goal of optimization is to minimize the objective function.

To be more specific, training error and generalization error are ordinarily different:

1. As the objective function of optimization algorithm is often a loss function, so its goal is to decrease the error in the training process.

2. In deep learning or statistical field, the goal is to reduce generalization error.

To accomplish the latter we need to pay attention to over-fitting in addition to using the optimization algorithm to reduce the training error [16].

There lots of problems when we try to optimize loss function of a deep learning model.

Therefore, in this section, we will focus on several controversial issues such as local minima, saddle point, gradient exploding and gradient vanishing.

As mentioned in loss function section, gradient-base method is mostly apply to train a deep learning model using backpropagation algorithm. Hence, in the process of training a model, the derivative calculation is done continuously and throughout to find the global minimum point of the function representing the data set - $f(x)$. For ease of visualization, we will look at figure 3.7.

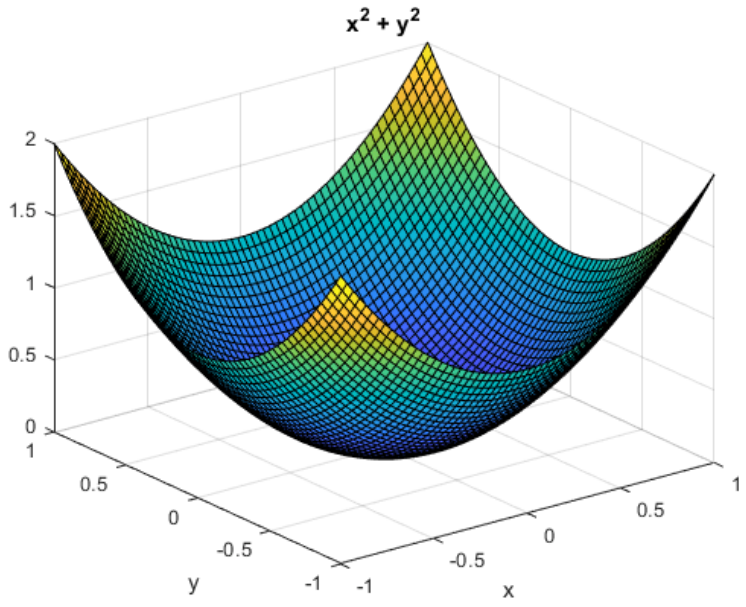


Figure 3.7: Bowl shape function - $x^2 + y^2$.

Assuming this cup-shaped plane is the equation f (x), its minimum point will be in the center of the cup. If we jump into this plane, of course we will slip straight to its minimum point. So when a function converges at its minimum, it optimizes as much as possible in the ability to classify the data. This is the fundamental goal of training: to continue modifying weights until the global minimum has been reached.

However, not all functions are cup-shaped. Let's take a look at figure 3.8 below for easier visualization.

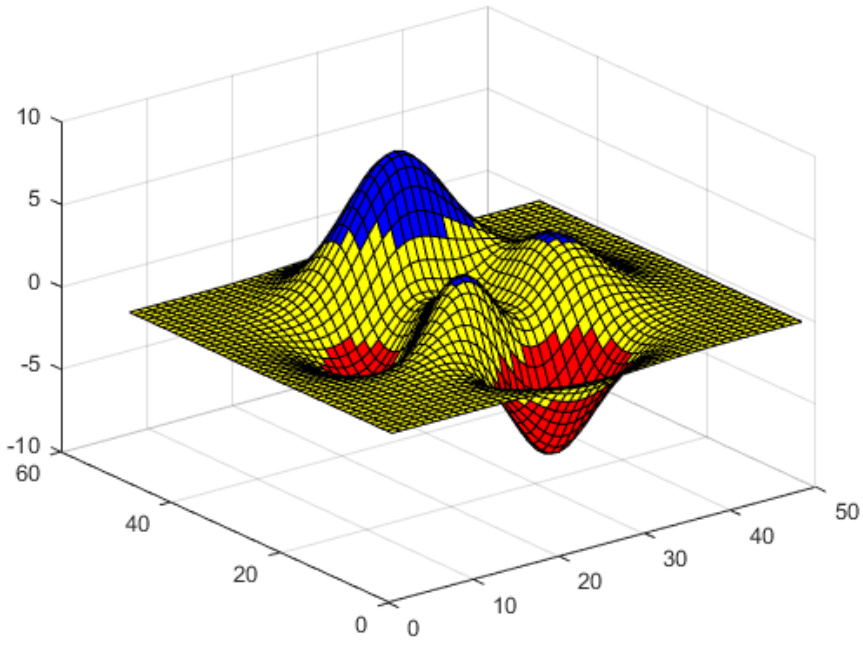


Figure 3.8: Local minima and global minima.

A function of the deep learning model usually takes the same form for many local minima points. Then, the function is susceptible to convergence at local minima points because it can only optimize the function of the model locally. This causes the gradients to be zero early and adversely affects the model's training and its results.

Besides local minima, saddle points are also a cause which make gradient vanish early. Although saddle point can cause gradient vanishing, it is not neither local minima nor global minima.

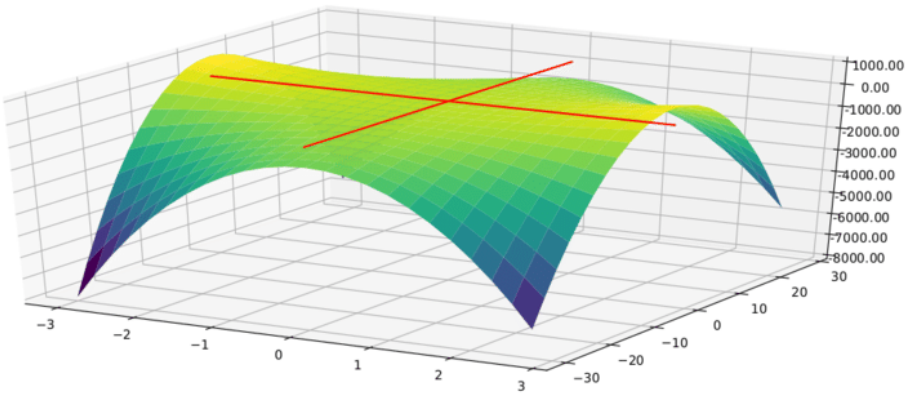


Figure 3.9: Saddle point.

Chapter 4

Region Based Convolutional Neural Network

4.1 The problem of region of interest

Computer vision is a large field that has been attracted lots of researchers, scientists in the recent years (after AlexNet's milestone). Beside image classification problem, another research trend of computer vision is object detection. The difference between two types of problem is that classification problem wants to know the label or labels of the image, while detection problem try to locate a bounding box surrounding the object inside the image. The challenge is there could be many bounding boxes representing different objects and we do not how many beforehand. The major reason why typical CNN models as we describe in section 2 cannot solve this problem is that the length of the output layer is not constant (unknown number of objects). A naive approach for this is that we can take different region of interest (RoI) and apply CNN model to check whether the RoI has object. Yet, this approach also has a huge problem is that the RoIs may have different spatial locations in the image, therefore, it will cost an expensive computational resource because of a huge number of RoIs.

4.2 The original R-CNN

In 2014, a new model had been proposed named region based convolutional neural network. The model has a remarkable record on PASCAL VOC 2010 with the mean

average precision (mAP) of 53.7% and mAP of 31.4% on ILSVR2013 detection dataset [3].

To solve the issue of the numerous Rois, Ross Girshick et al. proposed a method where we use selective search to extract just 2000 regions from the image and he called them region proposals [3]. As a result, instead of blowing up the computational resource to classify RoIs like the naive approach, now we can just work with 2000 regions.

According to the author, their system consists of three modules. The first module is the one that solve the expensive computation issue by using selective search to generate category-independent region proposals. The second module is a CNN that extract a 4096-dimensions feature vector from RoIs as the output. The last one is a set of SVMs and a linear regression models to classify the object and predict its bounding box, respectively.

Problems with R-CNN:

- Training is a multi-stage pipeline and expensive in space and time.
- It cannot be implemented as real time system because it takes 47s/image to extract features from RoIs when using VGG16 as backbone (on a GPU).
- The selective search algorithm is a fixed algorithm. Therefore, no learning is happening at that stage. This could lead to the generation of bad candidate region proposals.

4.3 The Improvement In Fast R-CNN And FASTER R-CNN

Next year, in 2015, Ross Girshick et al again proposed a new model called Fast R-CNN which solved the drawbacks of their previous model. This model architecture looks similar to the original R-CNN but instead of feeding the RoIs to CNN, an input image and multiple regions of interest (RoIs) are input into a CNN. Each RoI is pooled into a fixed-size feature map and then mapped to a feature vector by fully connected layers

(FCs). The network has two output vectors per RoI: softmax probabilitiesand per-class bounding-box regression offsets. The architecture is trained end-to-end with a multi-task loss [2].

Fast R-CNN has several advantages according to the paper:

- Higher detection quality (mAP) than R-CNN.
- Training is single-stage, using a multi-task loss.
- Training can update all network layers.
- No disk storage is required for feature caching.

With Fast R-CNN, we do not have to feed 2000 RoIs to CNN which enhance the training and testing speed by execute the convolution once per image and feature map will be generated from it. The network is illustrated in figure 6 (below).

Both of the above models (R-CNN & Fast R-CNN) use selective search to generate the RoIs. However, selective search is a slow and time-consuming process affecting the performance of the network [11]. Therefore, Shaoqing Ren et al came up with an object detection algorithm that alternates the selective search algorithm and allows the network completely learn to propose the RoIs and they called it Region Proposal Networks (RPNs). By sharing convolutional layers, the marginal cost for computing proposals is significantly decrease (e.g., 10s/image) [11].

As an upgrade from Fast R-CNN, the new model was introduced in 2016 by Shaoqing Ren et al called Faster R-CNN that improves the speed of the model towards real-time object detection. This new architecture is similar to its predecessor that image is provided as an input to a convolutional network to generate feature maps. Yet, instead of applying selective search, a separated network called RPNs (which is completely trained) is used to predict RoIs. In the next stage, the predicted results are reshaped using a RoIs pooling layer. And then, the rectangular RoIs are used for classifying the objects inside the image and predicting the offset value of bounding boxes.

Model	Describe	Testing speed	Limitations
CNN	Divides the image into multiple regions and then classify each region into various classes.	unknown	Naive approach take a huge number of regions to predict, therefore, expensive computation cost.
R-CNN	Apply selective search, choose 2000 regions to process separately.	47s on VOC07	Expensive computation cost from difference CNN models for each region
Fast R-CNN	Extract feature maps first then apply selective search to generate regions.	0.32s on VOC07	Selective search is slow and un-trainable.
Faster R-CNN	Alternate selective search by RPN which can be trainable and much faster.	0.2s on both COCO and VOC07	Object proposal takes time and as there are different systems working one after the other, the performance of systems depends on how the previous system has performed.

Table 4.1: Comparison between CNN, R-CNN, Fast R-CNN, and Faster R-CNN

4.4 Region proposal network (RPNs)

In object detection using region based technique, RPN is a one true backbone which takes an image as the input and outs a set of rectangular RoIs. Although the RPN can process a whole image, the ultimate goal of the author is to share computation with the Fast R-CNN. As the result, the RPN is modified to be a small network that works as the second stage in the model architecture and it take the convolutional feature map output by the last shared convolutional layer of Faster R-CNN object detection network as the input. And then, this small network slide a n x n spatial window over the feature map to map it from high dimensional feature to lower one. This feature then is fed to two siblings fully-connected layers - a box-regression layer and a box-classification layer [11].

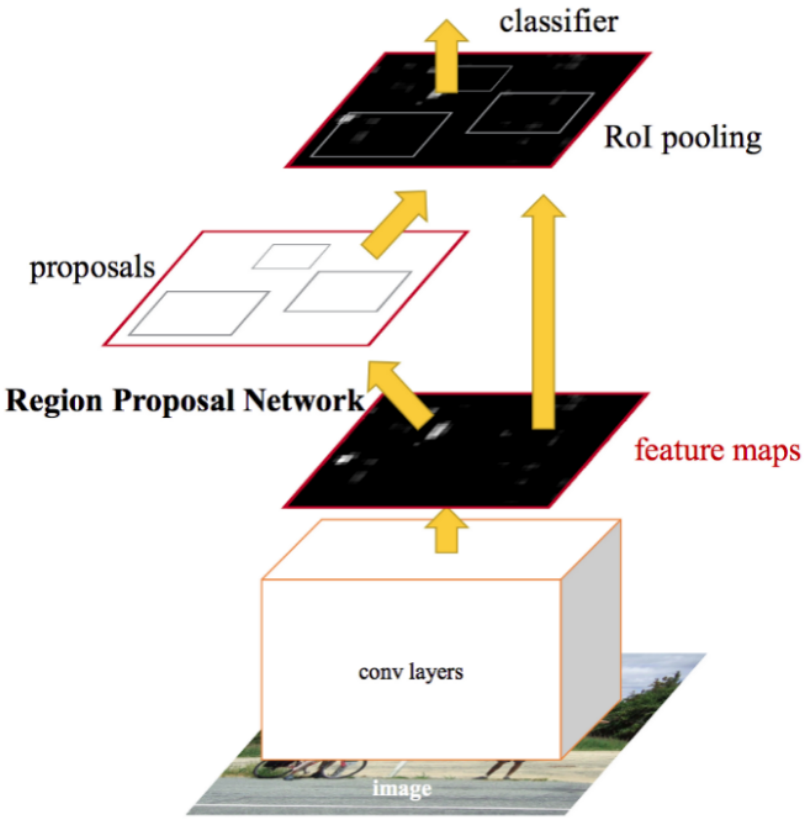


Figure 4.1: Faster R-CNN architect including RPN.

4.4.1 Anchors

At each sliding window position, we predict simultaneously lots of proposals with k is denoted as the maximum number of proposals at each location. According to the paper, the regression layer has 4k predicted coordinates of k boxes and classification layer has 2k predicted probability outputs of object or not object for each proposals. Therefore, the k value helps the authors parameterize the unknown number of boxes to k boxes and they called them anchors. An anchor is defined as the centered boxes of each sliding window. Base on the paper, the authors used 1 square, 2 rectangles with 3 scales and 3 aspect ratio to create 9 anchors (so k = 3 x 3 = 9). Therefore, with a feature map of the size W x H there are WHk anchors in total. These anchors are labeled with positive or negative base on the area that overlap with the ground truth box as the following rule:

- The anchor is positive when it is the maximum value of intersection over union (IoU) and it is bigger than 0.7.
- The anchor is negative when it is smaller than 0.3.
- The anchor that neither positive nor negative does not consider to be the training objective.

4.4.2 ROI pool layer

For each object proposal, RoI pooling layer extracts a fixed length of feature vector from the feature map so that it can be fed into the classifier and regressor in the final FC layer. More explicitly, the RoI pooling can be described in three steps:

- Dividing the region proposal into equal-sized sections (the number of which is the same as the dimension of the output).
- Finding the largest value in each section.
- Mapping these max values to the output buffer.

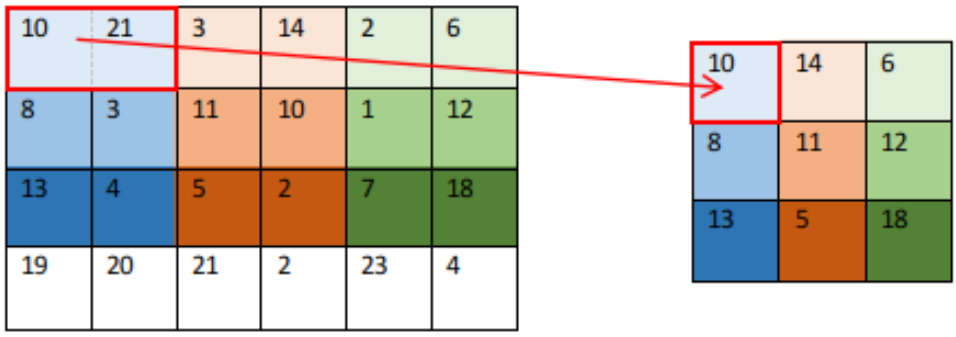


Figure 4.2: An example of RoI pooling window size 3x3 on vector 4x6.

As we can see on Figure 7, RoI pooling only take 1x2 vector to map its maximum value to a fixed length vector because quantization process ($4/3 \sim 1.3333 = 1$, $6/3 = 2$). Therefore, the entire bottom row does not take in to account causing missing information of the feature map. This process will directly affect the result accuracy.

Chapter 5

Mask R-CNN

5.1 The problem with Faster R-CNN

Until now, we know how Faster R-CNN extract feature maps by passing the image through lots of convolutional layer. However, while down sampling the image, we also scaled down the RoIs inside the image with a specific factor and round the offset of their bounding boxes. As a result, we create a new bounding box for the object inside which cause missing information and reduce performance of our system.

An example below illustrates when we feed 512x512 image to VGG16 and cause quantization on bounding box offset. The blue part is the missing piece data and the one green is a new data created by quantization.

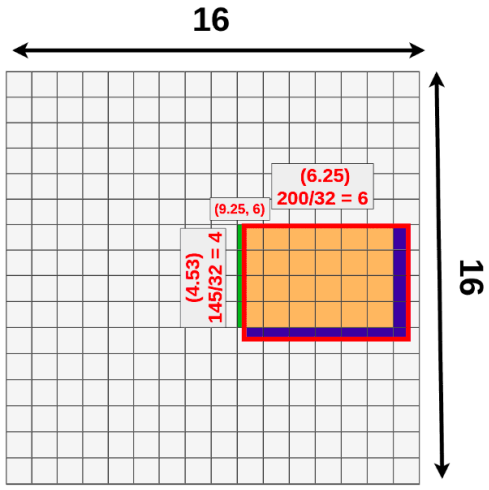


Figure 5.1: Example of quantization problem causing missing information.

As mention before in RoI pooling layer, RoI pooling operation will quantize floating number of RoI offset to a discrete offset. Then this quantized RoI is subdivided into spatial bins which are then aggregated (usually by max pooling) [5]. And once again, we lose vector information. We can look at Figure 7 for more details.

5.2 An extended solving the problem - Mask R-CNN

In 2017, a group of Facebook AI researchers - Kaiming He at el presented a new system which is influenced by Faster R-CNN called Mask R-CNN. This system, gennerally, is an extend of Faster R-CNN with a new branch for segmentation generating an object mask. Unlike others system which are complex multiple-stage cascade that predicts segment proposals from bounding-box proposals, followed by classification. Mask R-CNN allows three branches (including the existing branch for classification and bounding box regression, and a branch for predicting segmentation masks on RoI) to run in parallel which enhance the processing speed.

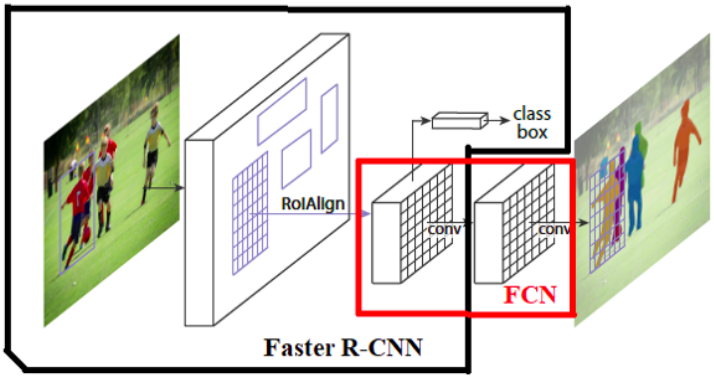


Figure 5.2: Mask R-CNN architecture including Faster R-CNN and FCN

As Figure 8 shows that Mask R-CNN is constructed by Faster R-CNN and a small fully convolution neural network. Because it is a stack of convolutional layer, the mask branch only consumes a small computational resource enabling the system to run extremely fast in testing phrase just like Faster R-CNN speed (0.2s/image).

It is noticeable that Mask R-CNN's authors solved the problem of RoI Pooling quantization by proposing a brand-new alternative technique called RoI Align which completely secure spatial locations of bounding box and information inside it. According to the paper, this minor change can affect the result accuracy up to 50%.

5.2.1 Multi-task loss

Inherit the spirit of Faster R-CNN, Mask R-CNN also has two stage procedure, with RPN is the first stage that generates proposals. In the next stage, in parallel with the original branch of Fast R-CNN, there is a mask generation branch that outputs a binary mask for each RoI. Therefore, a new loss function was proposed and defined as Ł = $L_{cls} + L_{box} + L_{mask}$. The classification and regression loss (L_{cls}, L_{box}) are the same as those defined in [2].

The mask loss, according to the authors, was defined as the average binary cross entropy loss (per-pixel sigmoid and binary loss) enabling the system to generate masks for each class without causing competition among them [4]. As a result, lots of masks will be generated but only the ground truth masks of class k in the corresponding RoIs are considered while calculating the mask loss (masks on another classes are not contribute to mask loss of class k). And by taking advantage of the classifier branch, the predicted class will be used to choose the output mask. This strategy decouples the mask and class prediction and outputs good instance segmentation results [4]. Therefore, Mask R-CNN's strategy is different from most techniques at that time (in semantic segmentation problems) when adopting a per-pixel multinomial logistic loss and validate with the standard metric of mean pixel intersection over union (IoU), with the mean taken over all classes - softmax (competition between classes), including background [4].

References

[1] Luiz Camargo, Hegler Tissot, and Aurora Pozo. Use of backpropagation and differential evolution algorithms to training mlps. pages 78–86, 11 2012.

[2] Ross Girshick. Fast r-cnn. 04 2015.

[3] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. 11 2014.

[4] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1, 06 2018.

[5] Thomas Heuvel, Dagmar de bruijn, Chris de Korte, and Bram Ginneken. Automated measurement of fetal head circumference using 2d ultrasound images. *PLOS ONE*, 13:e0200412, 08 2018.

[6] Salman Khan, Hossein Rahmani, Syed Shah, and Mohammed Bennamoun. A guide to convolutional neural networks for computer vision. *Synthesis Lectures on Computer Vision*, 8:1–207, 02 2018.

[7] Yann Lecun, Leon Bottou, Y. Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324, 12 1998.

[8] Jing Li, Yi Wang, Baiying Lei, Jie-Zhi Cheng, Jing Qin, Tianfu Wang, Shenli Li, and Dong Ni. Automatic fetal head circumference measurement in ultrasound us-

ing random forest and fast ellipse fitting. *IEEE Journal of Biomedical and Health Informatics*, PP:1–1, 05 2017.

[9] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. pages 3431–3440, 06 2015.

[10] Wei Lu, Jinglu Tan, and Randall Floyd. Fetal head detection and measurement in ultrasound images by an iterative randomized hough transform. *Ultrasound in medicine & biology*, 31:929–36, 08 2005.

[11] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. pages 1–10, 01 2016.

[12] Frank Rosenblatt. *The Perceptron: A Probabilistic Model for Information Storage and Organization (1958)*, pages 183–190. 02 2021.

[13] Joseph Walsh, Niall O' Mahony, Sean Campbell, Anderson Carvalho, Lenka Krpalkova, Gustavo Velasco-Hernandez, Suman Harapanahalli, and Daniel Riordan. Deep learning vs. traditional computer vision. 04 2019.

[14] Rikiya Yamashita, Mizuho Nishio, Richard Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 9, 06 2018.

[15] Zhao Yanling, Deng Bimin, and Wang Zhanrong. Analysis and study of perceptron to solve xor problem. pages 168 – 173, 12 2002.

[16] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*. 2019. `http://www.d2l.ai`.