الجمهورية الجزائرية الديموقراطية الشعبية

People's Democratic Republic of Algeria

Ministry of High Education and Scientific Research	وزارة التعليم العالي والبحث العلمي
University of Mohamed Khider, Biskra	جامعة محمد خيضر بسكرة
Faculty of Sciences and Technology	كلية العلوم والتكنولوجيا
Department of Electrical Engineering	قسم الهندسة الكهربائية
Ref.:	المرجع: .....

جامعة محمد خيضر بسكرة

Université Mohamed Khider Biskra

Thesis submitted for obtaining the degree of

Doctorate in: electronic engineering

Specialty (Option): Signals and communications

Image segmentation using convolutional neural networks (Application to dermoscopy)

Presented by:

HAFHOUF Bellal

Defended publicly on: 25/04/2023 **In front of the Jury composed of:**

Pr. BAARIR Zine-Eddine	President	Professor	Univ Biskra
Pr. ZITOUNI Athmane	Supervisor	Professor	Univ Biskra
Pr. MEGHERBI Ahmed Chaouki	Co-supervisor	Professor	Univ Biskra
Pr. MIMI Malika	Examiner	Professor	Univ Mostaganem
Pr. SBAA Salim	Examiner	Professor	Univ Biskra

Abstract

This PhD study is a part of the field of image segmentation and is particularly interested in the issue of automatic segmentation of skin lesions from dermoscopy images. Automated and accurate segmentation of skin lesions is an important step in Computer-Aided Diagnosis systems (CADs) for melanoma detection. Although numerous methods have been proposed in the literature, this task is still a challenging issue due to various factors related to the captured image of the skin, such as the presence of hair and blood vessels, low contrast of lesion and its surrounding healthy skin. Some lesions have fuzzy borders, wide variations in sizes and colors, and complex textures.

Recently, we have witnessed great success of using deep learning and especially convolutional neural networks (CNNs) in semantic segmentation and medical image analysis. To deal with the challenge of skin lesions segmentation, we propose models based on CNNs and specifically U-Net architecture (encoder-decoder). To demonstrate the robustness and effectiveness of the proposed methods compared to the state-of-the-art deep learning models, experimental results are reported on three datasets, including the IEEE International Symposium on Biomedical Imaging (ISBI) 2017, ISBI 2016, and PH2.

Keywords: Skin lesion segmentation, Dermoscopy, Deep Learning, CNN, Encoder–decoder, U-Net, Dilated convolution, Pyramid pooling, FCN, Atrous Spatial Pyramid Pooling.

Publications

- B. Hafhouf, A. Zitouni, A. C. Megherbi, and S. Sbaa, "An Improved and Robust Encoder–Decoder for Skin Lesion Segmentation," *Arabian Journal for Science and Engineering,* pp. 1-15, 2022.

- B. Hafhouf, A. Zitouni, A. C. Megherbi, S. Sbaa, and . "A Modified U-Net for Skin Lesion Segmentation," presented at the 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), EL OUED, Algeria, pp. 225–228, IEEE (2020).

Acknowledgments

First of all, I would like to thank Allah for giving me the ability to prepare this thesis. Praise to Allah, Lord of the worlds.

I would like to thank my supervisor Prof. Athmane Zitouni and my co-supervisor Prof. Ahmed Chaouki Megherbi for guiding me during the entirely of my PhD studies.

Also, I would like to thank all members of the jury (Pr. BAARIR Zine-Eddine, Pr. SBAA Salim, and Pr. MIMI Malika) for having agreed to read and evaluate the present manuscript.

Dedication

- To my family: my parents (father and mother), my brothers and sister, my uncles and aunts, my cousins (in particular **korichi fouad**)
- To my friends and colleagues

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Introduction

Malignant melanoma is the most deadly type of skin cancer, and its incidence rate has been steadily increasing in the world for the past decade [1, 2]. For example, in the USA for 2023, approximately 97.610 new cases of melanoma will be diagnosed, and about 7.990 people are expected to die [3]. However, early diagnosis and careful treatment can increase the survival rate. Dermoscopy is one of the most popular techniques in the early diagnosis of melanoma. Dermoscopy is a non-invasive imaging tool that provides a magnification of the images, and allows a better visualization of deep skin structures when compared to conventional clinical images [4]. Accurate analysis of dermoscopy images using the naked eye alone is time-consuming, complex, subjective, and not reproducible. Therefore, Computer-Aided Diagnosis systems (CADS) can be used to help dermatologists for melanoma detection [5]. These tools are generally comprised of the four following steps namely: image acquisition, lesion segmentation, feature extraction, and finally the classification.

1.2 Problematic

Lesion segmentation is the process of isolating skin lesion from its surrounding healthy skin (background). It is an important task since it affects the accuracy of the subsequent steps of the diagnosis system (CADS) [6, 7]. Automatic segmentation of skin lesions is a challenging task due to various factors related to the captured image of the skin, such as the presence of hair and blood vessels, low contrast of lesion and its surrounding skin, the fact that some lesions have fuzzy borders, vary in sizes and colors, have complex textures. Some of these challenging situations are illustrated in Fig. 1.1.

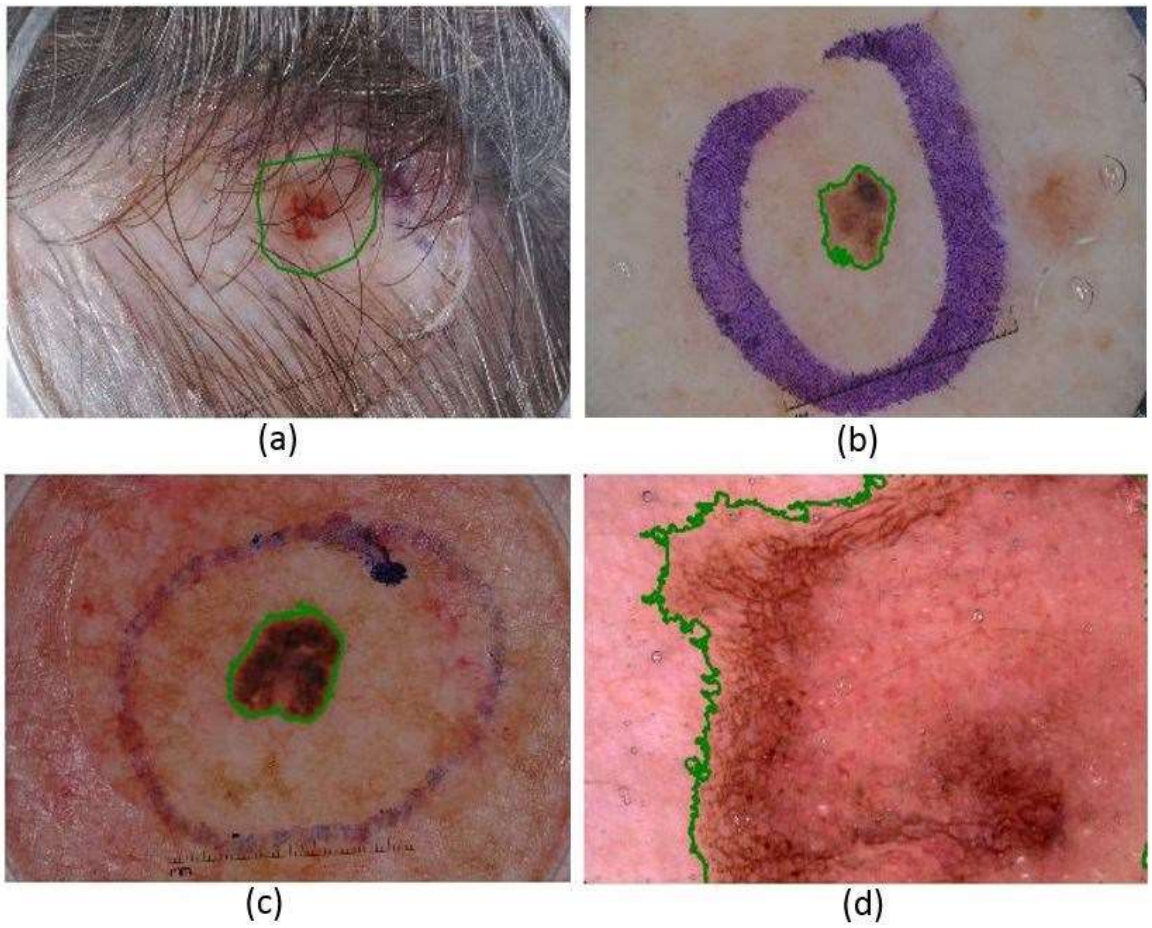

Figure 1.1. Some challenging examples of ISBI 2017 dataset, such as low contrast (a, b, c, d), presence of hair (a), ink marks (b, c), and irregular boundaries (b, d). Green contours represent the lesions segmented by dermatologists

Existing unsupervised techniques, such as thresholding, clustering, edge-based, region-based and classical supervised methods [8] rely on low-level handcrafted features (appearance information). Consequently, these traditional methods have limitations and fail to solve the challenging situations, such as low contrast and the presence of hair [5, 9, 10]. Additionally, these methods require pre-processing steps, such as illumination correction and hair removal.

Recently, methods based on deep learning, especially convolutional neural networks (CNNs), have achieved great success in computer vision and medical image analysis (classification, detection, segmentation, etc.) [11, 12]. This success is mainly due to the

availability of large amount of labeled data (ImageNet [13]...), powerful computational resources, such as graphic processing units (GPUs), and open source libraries and frameworks (Tensorflow, Keras, PyTorch, etc.). Deep learning based methods have been applied to the skin lesion segmentation task and outperformed the traditional methods. This is due to the fact that deep learning methods can extract low-level local information as well as high-level semantic information in a hierarchical representation from the input images. However, in terms of processing time, these methods are computationally expensive, even with using a powerful hardware (GPU). The second issue is the generalization capability (the capability of a trained model to yield good results on an independent dataset).

Therefore, the main objective is to build an accurate, fast, and robust deep learning based segmentation model to assist the subsequent steps of the CAD system for melanoma detection.

1.3 Thesis Contributions

In this thesis, three deep learning approaches based on the state-of-the-art U-Net architecture [14](encoder-decoder) are proposed to segment skin lesions from dermoscopic images. In the first approach [15], we use dilated convolution [16] and pyramid pooling modules (PPM) [17] to enhance the segmentation results. This approach is tested on the ISBI 2016 dataset and compared to U-Net as a baseline. In the second approach, we propose a novel fully convolutional network called FCN-MASPP. The novel module MASPP (modified atrou spatial pyramid pooling), which is inspired from DeepLab [18-20] and PSPNet [17], is used at every level in both the encoder and the decoder. We compare the results of FCN-MASPP on the ISBI 2017 dataset to the equivalent structures (a basic FCN and U-Net). The third approach [21] is the main contribution of the thesis. In this approach, we propose an improved scheme that adopts as the encoding path (encoder): 10 standard convolutional layers, followed by a pyramid pooling module (PPM) and a dilated convolutional block (DCB). The dilated residual blocks (DRBs), which consist of dilated convolutional layers with residual connections, are introduced in the decoding path (decoder) to further refine the segmentation maps. So the main contributions of this approach can be summarized in these points:

1. Proposing an encoding path, which consists of the first 10 convolutional layers of the VGG16 network [22], followed by a pyramid pooling module (PPM) and a dilated convolutional block (DCB). This combination enables more representative of extracted feature maps and preserves more spatial resolution.
2. Instead of using conventional convolutional layers like U-Net, we introduce dilated residual blocks (DRBs) in the decoding path to extract more context information for dense prediction.
3. We experiment on three public datasets, including ISBI 2017, ISBI 2016, and PH2.
4. We compare the performance of our proposed model against state-of-the-art models FCN [23], SegNet [24], U-Net [14], and U-Net ++ [25] as baselines, and other recently published methods.

1.4 Thesis outline

This thesis is organized as follows. **Chapter 2** discusses the background concepts for deep learning, including supervised machine learning, artificial neural networks (ANNs), and their training strategy. Deep convolutional neural network architectures (DCNNs) and some popular state-of-the-art CNNs based semantic segmentation models are introduced in **chapter 3**. **Chapter 4** will focus on Skin lesion segmentation methods for dermoscopy images. **Chapter 5** presents experimental results. Finally, **chapter 6** concludes the thesis and discusses some future works.

Chapter 2
Background concepts for Deep Learning

2.1 Introduction

Deep learning (DL) is a subfield of machine learning (ML). It attempts to learn automatically multiple levels of representations with multiple levels of abstraction from raw input data [26]. It is basically built on deep neural networks. The key aspect of deep learning is that it skips the feature extraction step designed by human engineers [26, 27]. As illustrated in Fig. 2.1, the model takes raw input data (images, for example), and the learned features of the lower level layers detect trivial patterns in the input data (i.e. lines, edges). These elementary detected patterns are used to encode more complex features (i.e. corners, contours). Finally, the features of the next level layers operate with the most abstract representations combining the obtained corners and contours into object parts [27].

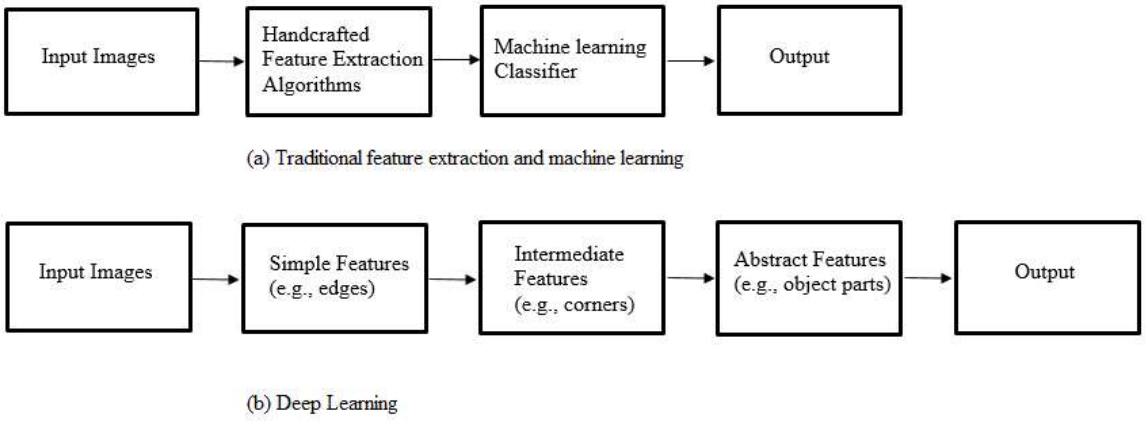

Figure 2.1. Machine Learning (a) and Deep Learning (b) approaches. From [27]

In the following, we give a brief introduction to machine learning and supervised learning. Then, we detail the artificial neural networks (ANNs), the training methodology, and some regularization techniques.

2.2 Machine learning and supervised Learning

Machine Learning (ML) is a subfield of artificial intelligence (AI) aiming to give computers the ability to learn from data. Tom Mitchell has described that the field of ML is concerned with the question of how to construct computer programs that automatically improve with experience [28]. There are different types of learning in ML systems:

Supervised learning, Semi supervised learning, Unsupervised learning, and Reinforcement learning. This thesis will only focus on supervised learning.

In Supervised Learning (SL), the model (computer) is presented with training examples and their corresponding labels (Fig. 2.2).

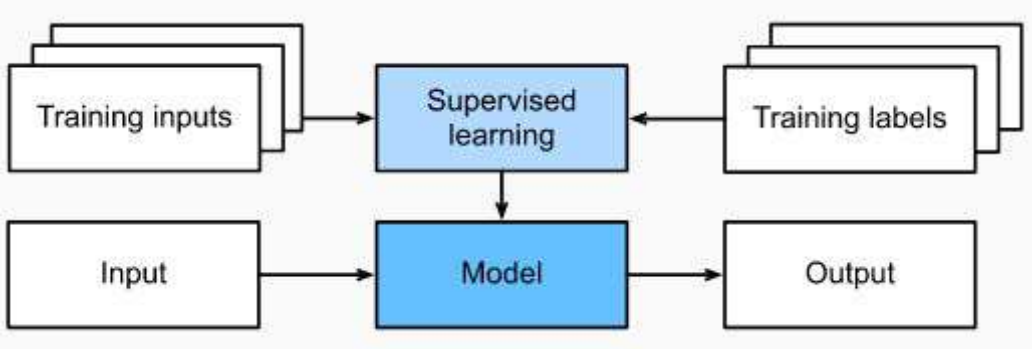

Figure 2.2. Supervised learning. From [29].

Let's assume a training dataset $\boldsymbol{\mathcal{D}} = \{(x^{(1)}, y^{(1)}), \ldots (x^{(n)}, y^{(n)})\}$

$x^{(i)}, y^{(i)}$ are the input and the desired output (label) of the $(i\text{-}th)$ example, respectively. The output label can be either continuous or discrete. For the first case, we talk about regression, while the second case is called classification. A training phase is performed to learn a model/function (from labeled training data) that best predicts inputs:

$$\hat{y} = f_{\Theta}(x) \tag{2.1}$$

$\hat{y}$: Denotes the predicted label by the model.

Θ: Denotes the model parameters to find by minimizing a loss function. This function, which is also called the 'cost function', measures the error between the predicted output and the desired (true) label.

After the training phase, the learned model can be used to measure its generalization ability on separate (unseen) data. This is called test phase. In conventional ML, features are first extracted from raw input data (e.g., images, videos, sounds, etc.) to transform it in some sense. Then these features are used for model learning, while DL aims at learning automatically features representation from raw data.

2.3 Artificial neural networks

Artificial Neural Networks (ANNs) are a set of algorithms inspired from the biological neurons, which are the nerve cells composing the human brain. Artificial neurons are arranged in layers and linked by weights in a similar way to synapses linking biological neurons.

2.3.1 Biological neuron

The biological neuron is shown in Fig. 2.3. It consists of:

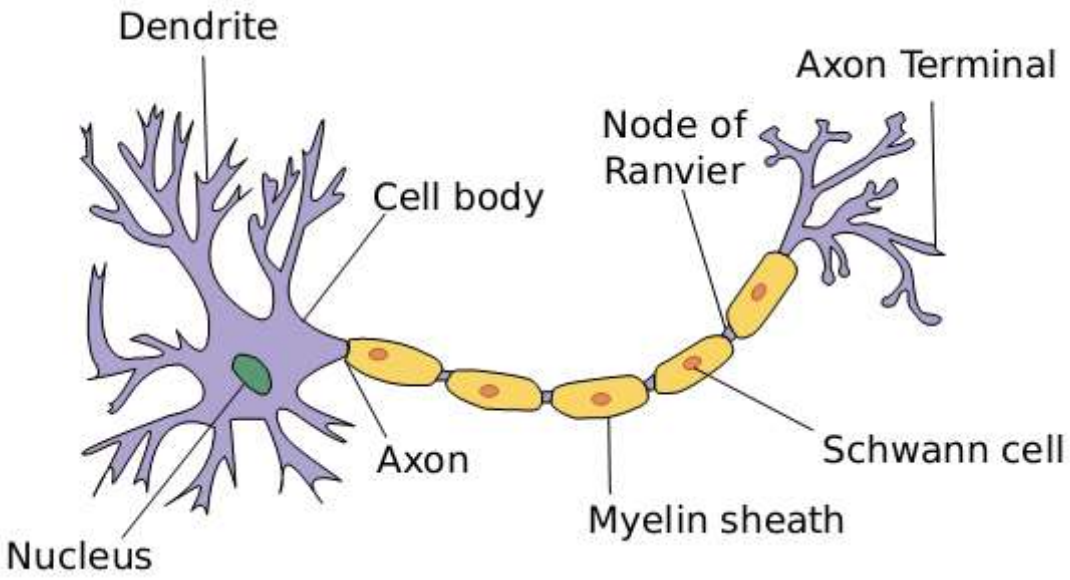

Figure 2.3. Biological neuron. From [29]

- Dendrites (input terminals): receive input information from sensors or other neurons.

- Cell body (also called soma): contains the nucleus of the neuron, which processes the information.
- Axon: carries the produced information to the axon terminal (synapse).
- Axon terminals (output terminals or synapses): are the transition between the axon and dendrites of other neurons.

2.3.2 Artificial neuron

In 1943, McCulloch and Pitts [30] proposed the first mathematical model of the biological neuron. This model performs a dot product with the input (x_i) and its weights(w_i), then adds the biais b and applies an activation function f, as can be described with the following equation:

$$y = f(\sum_i w_i x_i + b) \tag{2.2}$$

This first model (original) did not learn (the weights are random numbers), f is just a $sign$ (threshold) function. The artificial neuron is illustrated in Fig. 2.4.

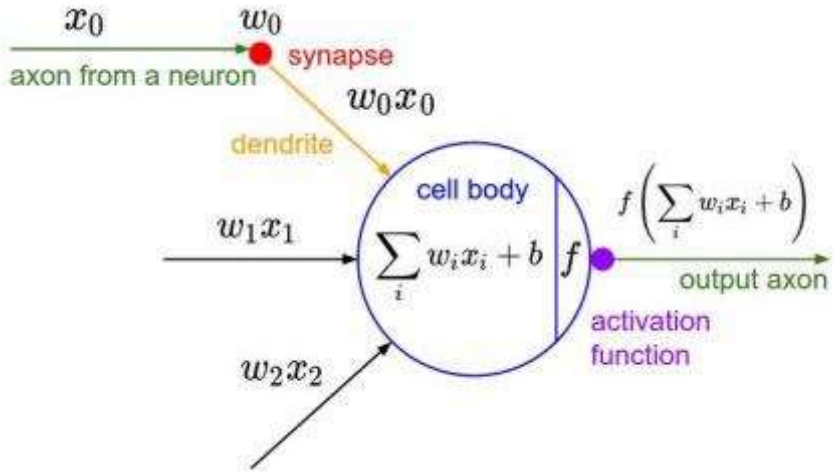

Figure 2.4. Artificial neuron. Extracted From [31]

2.3.3 Perceptron

In 1958, Rosenblatt [32] introduced the learning algorithm based on the artificial neuron, and called perceptron. The algorithm is the following:

Perceptron algorithm.

Initialize the weights (w_i) and biais (b) to small random numbers.

While $(i \leq \text{number of iterations}$ or predefined error threshold) do

Calculate the output of the classifier (model) using the equation (2.2).

Update the weights according to: $w^{(t+1)} = w^{(t)} + \eta(y - \hat{y})x$

end

In 1969, Minsky and Papert [33] published the limitations of the perceptron and demonstrated that the logical gate XOR cannot be expressed with one layer perceptron, which always performs a linearly separable problem. To overcome this limitation, the model was extended to Multi-Layer Perceptron (MLP) [34].

2.3.4 Multilayer Perceptron (deep neural networks)

An MLP is a deeper network based on the perceptron unit. It consists of three types of layers where each neuron in a layer is fully connected to each neuron in the subsequent layer:

An input layer, where its neurons are fed with the input data.

An output layer, which corresponds to model outputs. For multi-class classification, the $softmax$ function can be used to compute the final predictions (class probabilities) as follows:

$$softmax(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{2.3}$$

Hidden layers (intermediate layers between input and output layer). Each one consists of multiple neurons. If a multilayer perceptron contains multiple hidden layers, it will be called deep neural network, hence the term “deep learning”. An illustration of a multilayer perceptron with two hidden layers is shown in Fig. 2.5.

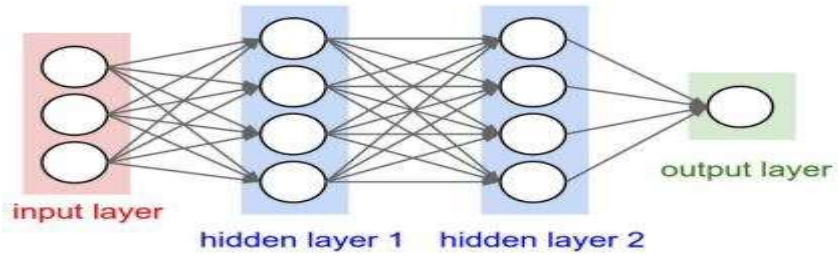

Figure 2.5. An example of a multilayer perceptron [31]. Each neuron in a layer is connected to each neuron in the next layer.

Formally, we calculate the output of each neuron (called activation) in a hidden layer as follows:

$$x_i^{(l+1)} = f\left(\sum_{j=1}^{n} w_{ij} x_j^{(l)} + b_j\right) \tag{2.4}$$

Where $x_i^{(l+1)}$, $x_j^{(l)}$ are the activation of neuron i of the $(l+1)$-th layer and the activation of neuron j of the (l)-th layer, respectively. w_{ij} are the weights associated to the neuron $x_i^{(l+1)}$. n indicates the number of neurons in the (l)-th layer.

In MLP, to perform non-linearity to the network, non-linear activation functions are used, instead of a $sign$ function. Some of these activation functions are illustrated in Fig. 2.6, including the hyperbolic tangent($tanh$), $sigmoid$ and rectified linear unit ($ReLU$). $ReLU$ is widely adopted rather than the other activation functions, as it reduces the vanishing gradient and accelerates the training process.

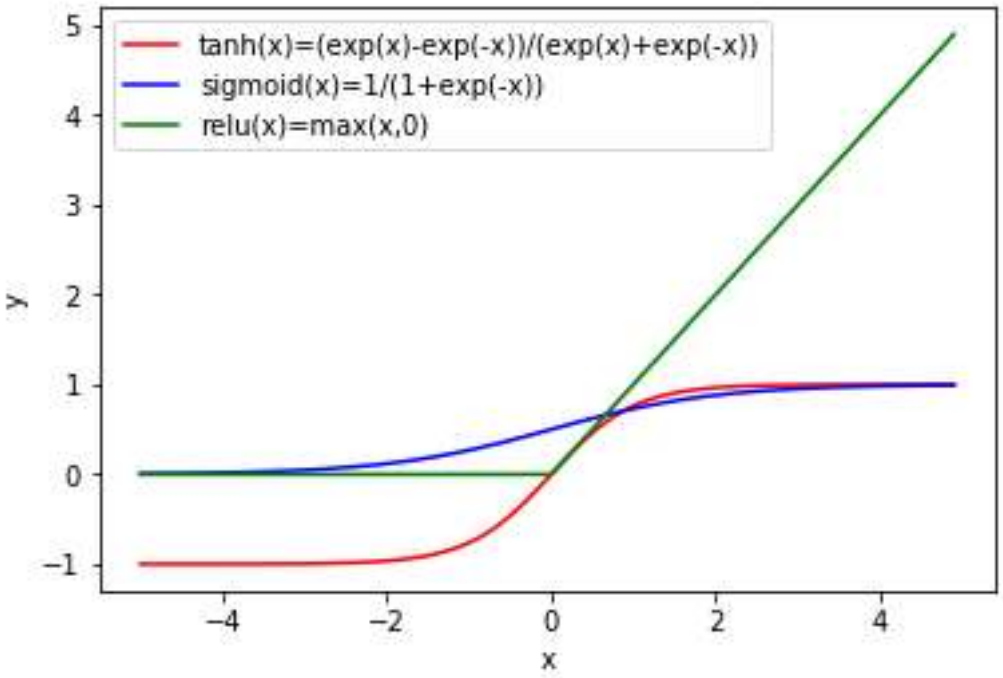

Figure 2.6. Some nonlinear activation functions.

2.3.5 Training ANNs

The goal of the training process is to estimate the model parameters that best predict the input by minimizing a loss function (error) $L(y,\hat{y})$. This error can be any differentiable function measuring the mismatch between the predicted outputs by the model and the true labels. For example, binary cross entropy (BCE) (equation 2.5), categorical cross entropy (CCE) (equation 2.6), and mean square error (MSE) (equation 2.7) are used for two-class classification, multi-class classification (with K classes), and regression problems, respectively.

$$BCE = -\sum_{i=1}^{N}(y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)) \tag{2.5}$$

$$CCE = -\sum_{c=1}^{K} y_{c,i} \log \hat{y}_{c,i} \tag{2.6}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.7}$$

Before starting the training process, all parameters are randomly initialized. In addition to the loss function, other hyperparameters are established, such as the activation function, number of layers, etc. The process of learning used for adjusting the model parameters (weights) is referred to as the backpropagation method [35].

2.3.5.1 Backpropagation and stochastic gradient descent

The backpropagation algorithm starts at the last layer by computing the error between the predicted output by the model (during the forward pass) and the expected output (true label). Then, this error is propagated back through the network towards the inputs. For each layer (during the backward pass), the gradients of the error are computed using the chain rule, and applied using an optimization algorithm such as stochastic gradient descent (SGD) to update the model parameters (weights) (Fig. 2.7). SGD is the process of randomly extracting a small subset called mini-batch of size m from the entire training set, computing error and then the gradients to update the parameters Θ as

$$E = \frac{1}{m}\sum_{i=1}^{m} L(y_i, \hat{y}_i) \tag{2.8}$$

$$\Theta = \Theta - \eta\frac{\partial E}{\partial \Theta} = \Theta - \eta\nabla E \tag{2.9}$$

$\Theta = \{W, B\}$, where W is a set of weights and B a set of biases.

η is a hyperparameter called the learning rate, which controls the speed of the training process. It is often thought of as one of the most important hyperparameters that we will have carefully tune [36]. A learning rate that is too large can cause an oscillation around the suboptimal solution; whereas the too small value causes slow convergence (see Fig. 2.8).

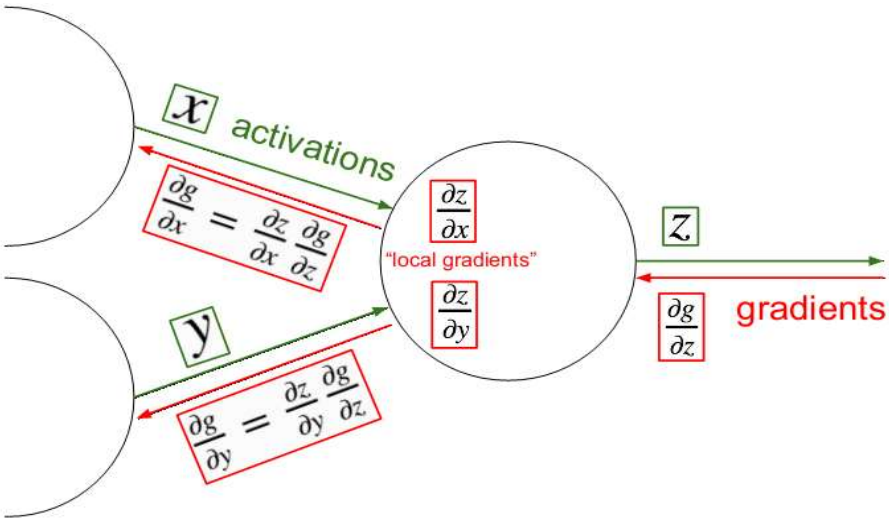

Figure 2.7. An illustration of backpropagation. From [31]. The green direction indicates the forward pass (prediction), while the red direction indicates the backward pass, where the gradient is computed using the chain rule and propagated back to update the weights.

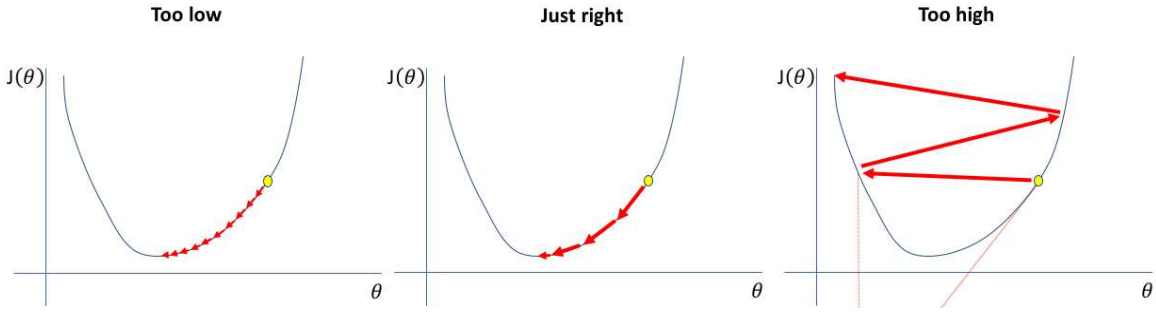

Figure 2.8. Effect of learning rate. From[37]

The training process takes several epochs to allow the model to learn and finally converge. An epoch is when the entire training dataset is passed forward and backward through the neural network exactly once. The number of mini-batches needed to complete one epoch is called iterations.

In the following, we will outline some of the other sophisticated optimization algorithms, including Momentum, Adagrad, RMSProp, and Adam.

Momentum

Momentum [38] is a variant of SGD, designed to reduce the noise produced by SGD and accelerate the convergence by adding a fraction of the previous update to the current update. Introducing momentum yields the following update equations:

$$v = \gamma v - \eta \nabla E \tag{2.10}$$

$$\Theta = \Theta + v \tag{2.11}$$

v is known as velocity. It accumulates movement in the direction of the minimum, thus leading to faster convergence. γ is a momentum coefficient and commonly set to 0.9.

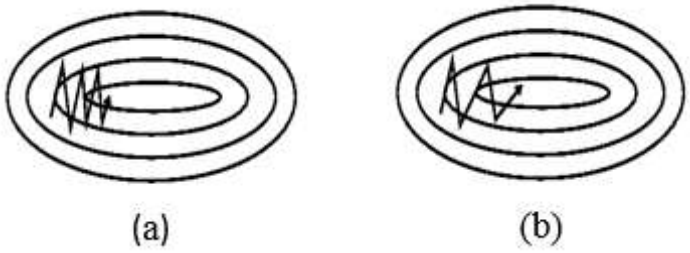

Figure 2.9. SGD. (a) without momentum. (b) with momentum that reduces oscillations. From [39]

Adagrad

In SGD, the learning rate is the same for all parameters and for each iteration. Adaptive Gradient [40], another variant of SGD, adjusts the learning rate for each parameter by using the gradients from previous iterations (an accumulation of the historical gradients), allowing larger updates for less frequent parameters and smaller updates for frequent parameters. The update equations in Adagrad are

$$\Theta = \Theta - \frac{\eta}{\sqrt{s+\epsilon}} \nabla E \tag{2.12}$$

$$s = s + (\nabla E)^2 \tag{2.13}$$

s is the sum of the previous squared gradients until the current iteration. ϵ is used to prevent division by zero.

RMSProp

The problem with Adagrad is the decaying learning rate, which becomes infinitesimally small after a considerable number of iterations, and eventually no progress can be made (the model can not learn new knowledge). To deal with this problem, RMSProp (Root Mean Squared Propagation) [41] restricts the number of previous gradients to some fixed size rather than using the full set of previous gradients. In RMSProp optimization algorithm, the learning rate is computed using an exponential average of past squared gradients instead of a naïve accumulation of past squared gradients. The updates to the gradient accumulation and parameters are as follows:

$$s = \gamma s + (1 - \gamma)(\nabla E)^2 \tag{2.14}$$

$$\Theta = \Theta - \frac{\eta}{\sqrt{s+\epsilon}} \nabla E \tag{2.15}$$

Adam

Adam (adaptive moment estimation) [42] is another optimization algorithm that its learning rate changes adaptively for each parameter. This algorithm is currently one of the most popular and preferred optimizer for deep learning. Adam is a combination of Momentum and RMSProp. It estimates the first moment of the gradient v (similar to Momentum) and the second moment of the gradient s (similar to RMSProp) using exponential moving average:

$$v = \beta_1 v + (1 - \beta_1)(\nabla E) \tag{2.16}$$

$$s = \beta_2 s + (1 - \beta_2)(\nabla E)^2 \tag{2.17}$$

β_1, β_2 are the decay rates of the first moment and the second moment of the gradients, respectively. However, these estimations are biased towards zero, especially in the initial steps for small decay rates. At each iteration t, the following equations can be used to remedy this bias:

$$\hat{v} = \frac{v_t}{1-\beta_1{}^t} \tag{2.18}$$

$$\hat{s} = \frac{s_t}{1-\beta_2{}^t} \tag{2.19}$$

Finally, the update rule for Adam is as follows:

$$\Theta = \Theta - \frac{\eta}{\sqrt{\hat{s}+\epsilon}}\hat{v} \tag{2.20}$$

2-3-6 Reduce overfitting in ANNs

When the model performs well on the training dataset, but poorly on an unknown testing dataset, this problem is called overfitting. This issue can occur when complex networks are used to solve simple problems. Several regularization techniques are then used to mitigate the overfitting issue and improve the generalization of the model. Some of the techniques we will use in our experiments include early stopping, dropout [43], batch normalization [44], and data augmentation.

Early stopping

In practice, we split the dataset into a training set and a validation set. The training set is used to update the model parameters (weights), while the validation set is used to determine the model performance. During training, we measure the error on training and validation sets, and when the error on an unseen validation set starts to increase (the model begins to overfit), then the network training is stopped and thus the computational cost of the training process is reduced. This strategy is known as Early stopping (see Fig. 2.10).

Dropout

Dropout layer is commonly used to reduce overfitting. During training, this layer randomly drops some neurons and their connections with a probability p, which ranges from 0 to 1. At test time, all dropped neurons will be active. This concept is depicted in Fig. 2.11.

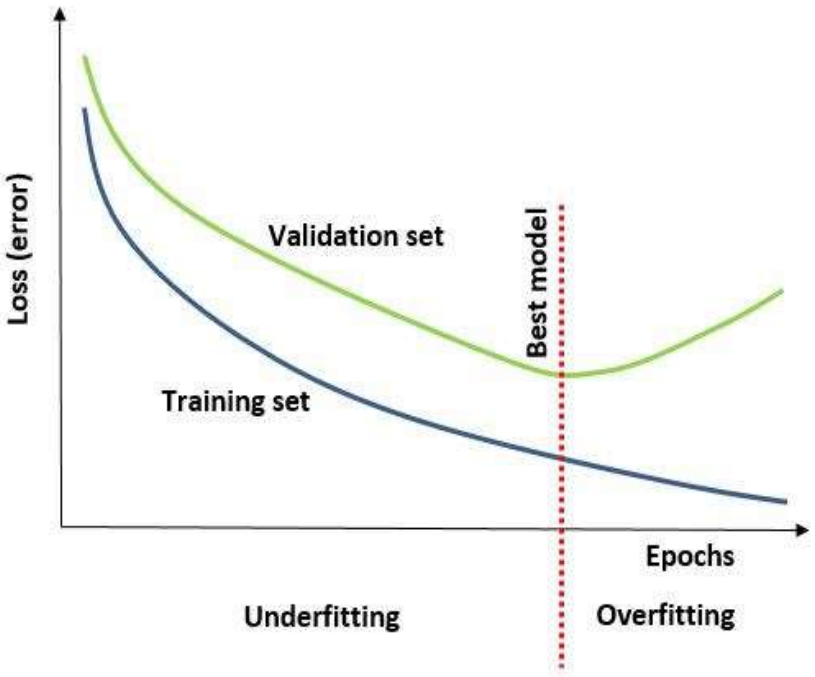

Figure 2.10. Overfitting and Early stopping strategy

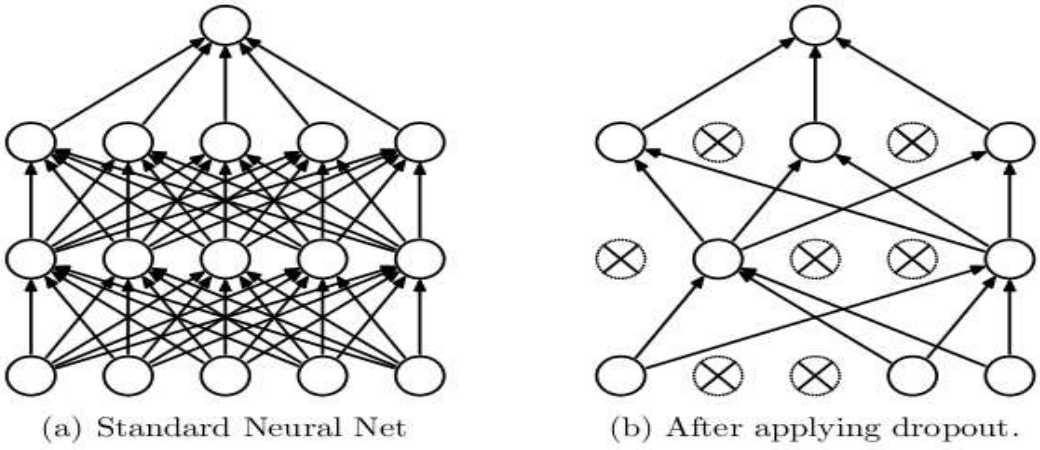

Figure 2.11. Dropout. From [43]. During training phase, some neurons are disactivated

Batch normalization

Batch normalization helps gradient propagation in the model training and accelerates the learning process. Also, it is another technique, which can be used as a regularizer to reduce overfitting. It normalizes the inputs to a layer for each mini batch to reduce the internal

covariate shift (change in the distribution of internal nodes during training). The procedure of batch normalization is described as follows:

1- Normalize the batch $B = \{x_1, x_2, \dots x_m\}$ using the mean and the variance of the batch calculated as:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{2.21}$$

$$\sigma_B{}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{2.22}$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B{}^2 + \epsilon}} \tag{2.23}$$

Where μ_B, $\sigma_B{}^2$, $\hat{x}_i$ are the batch mean, the batch variance, and the normalized values, respectively.

2- Apply linear transformations to the normalized values $\hat{x}_i$:

$$y_i = \gamma \hat{x}_i + \beta \tag{2.24}$$

γ and β are learnable parameters that correspond to the scaling and shifting in the transformation.

Data augmentation

Data augmentation is often used to augment the training samples using various transformation. For example, for image data, various geometric transformations such as rotation, flipping, and zooming might be used. The goal is that at training stage, the model will never see the exact same picture twice, and this helps the model to generalize better [45].

2.4 Deep learning architectures

There are several deep learning techniques currently used in several domains (speech recognition, robotics and control, medical imaging, bioinformatics, natural language processing, etc.). These include unsupervised learning such as auto-encoders (AEs), deep belief networks (DBNs), and generative adversarial networks (GANs); and supervised

learning such as recurrent neural networks (RNNs), long short-term memory (LSTM), and convolutional neural networks (CNNs). The following chapter presents CNNs, which are used through this thesis.

2.5 Conclusion

In this chapter, we presented a basic introduction to artificial neural networks (ANNs) and deep learning, which are essential for the understanding of later chapters. We described the artificial neuron (the first mathematic model of the biological neuron), the perceptron (the learning algorithm based on the artificial neuron), and the multilayer perceptron (a deeper network based on the perceptron unit). The process of learning in a multilayer perceptron is achieved using the backpropagation algorithm based on SGD optimization method or its variants such as the well-known Adam. Also, we presented some regularization techniques to reduce the overfitting. For further details and extensive presentation of the field, the readers can refer to [46] [47].

Chapter 3

Convolutional neural networks for semantic image segmentation

3.1 Introduction

Semantic segmentation, also called (pixel-wise classification or dense prediction) aims to assign a class (category) label to each pixel in an image. It is a common task and one of the key problems in natural images for visual scene understanding, and medical image analysis for lesion assessment and disease diagnosis. In this chapter, we present some popular state-of-the-art CNN based semantic segmentation models. First, we present a convolutional neural network (CNN) with some popular architectures. The typical usage of CNN is the image classification task. Then, we describe how to make this network suitable for segmentation.

3.2 Deep Convolutional Neural Networks (DCNNs)

CNNs [48] are a particular class of ANNs, where the networks perform convolution operations instead of matrix multiplication. CNNs are used for processing data with a spatial grid-like topology, such as images, which can be thought of as a 2D or 3D grid of pixels. The inspiration of the CNNs comes from the brain's visual cortex [49], where each neuron only responds to stimuli around a limited region of the visual field known as the receptive field. These networks have achieved great success in computer vision and medical image analysis. A schematic representation of a typical CNN is shown in Fig. 3.1. It contains subsequent layers of convolution (with activation functions) and pooling operations, and a fully connected layer.

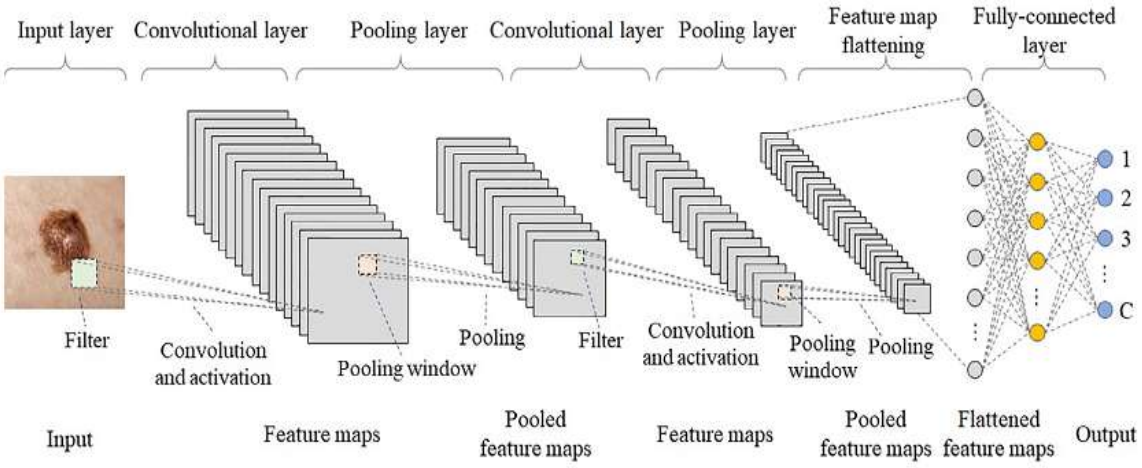

Figure 3.1. A typical CNN. From [50]

3.2.1 Convolutional layers

A convolutional layer is a set of convolutional kernels. Each kernel is convolved with an input tensor or a set of feature maps to produce a feature map. Given an input image I of size $H \times W \times D$, a kernel K of size $N \times N \times D$; where H, W, N are spatial dimensions, and D is the channel dimension. So, the convolution kernel is run along all the pixels in the input image, multiplying the surrounding pixel values with the kernel and adding them. The resulting output feature map $O = I * K$ ($*$ denotes convolution) is defined as:

$$O(i,j) = \sum_{k=0}^{D} \sum_{a=0}^{N} \sum_{b=0}^{N} I(i+a, j+b, k) K(i,j,k) \quad (3.1)$$

Every convolutional layer is usually followed by an activation function to perform nonlinearity to the network. Some of these activation functions are already described earlier like the most popular $ReLU$. Unlike a fully connected network (MLP) where each input neuron is fully connected to each output neuron in the previous layer, CNN neurons have the so-called local (sparse) connectivity. This means that each neuron only depends on a spatially local subset of inputs in the previous layer (each neuron only has a local receptive field). By exploiting a local substructure within the image, features that are more representative can be gradually learnt. Limiting the number of connections of each neuron means that computing the output requires fewer operations [47]. Another particularity of a CNN is the weight sharing that refers to using each kernel (smaller than the input image) with its fixed weights across different positions of the entire input image. Weight sharing allows reducing the number of learnable parameters (weights) and thus building a deeper network with fewer parameters.

3.2.2 Pooling layers

The pooling layers downsample the extracted feature maps by convolutional layers and thus increase the receptive field of the network. The commonly used one are 2×2 $max-pooling$ and $average\ pooling$. $Max-pooling$ only retains a pixel with the maximum value among the neighboring four pixels, while the $average\ pooling$ calculates the average value instead of just choosing the maximum value. The pooling layer has no learnable parameters and is useful for controlling the number of parameters and overfitting.

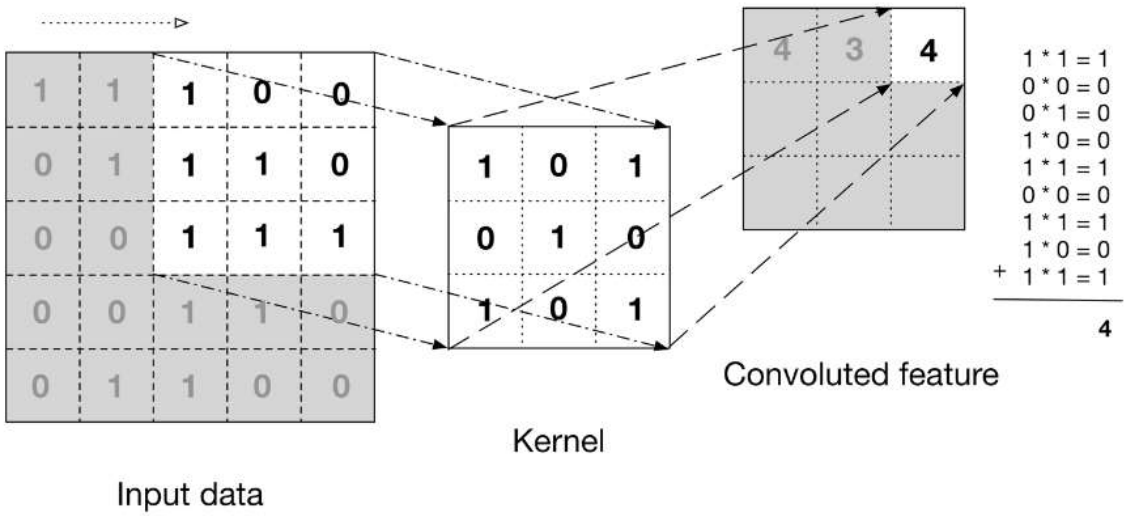

Figure 3.2. Convolution operation. From [51]

Furthermore, the pooling layer has the propriety of invariance to small translations, which is useful when a feature is present is more important than its precise location.

3.2.3 Fully connected layer

The last layer in a CNN is the fully connected layer. Similar to an ANN, each neuron in the fully connected layer is connected to all of the neurons in the previous layer.

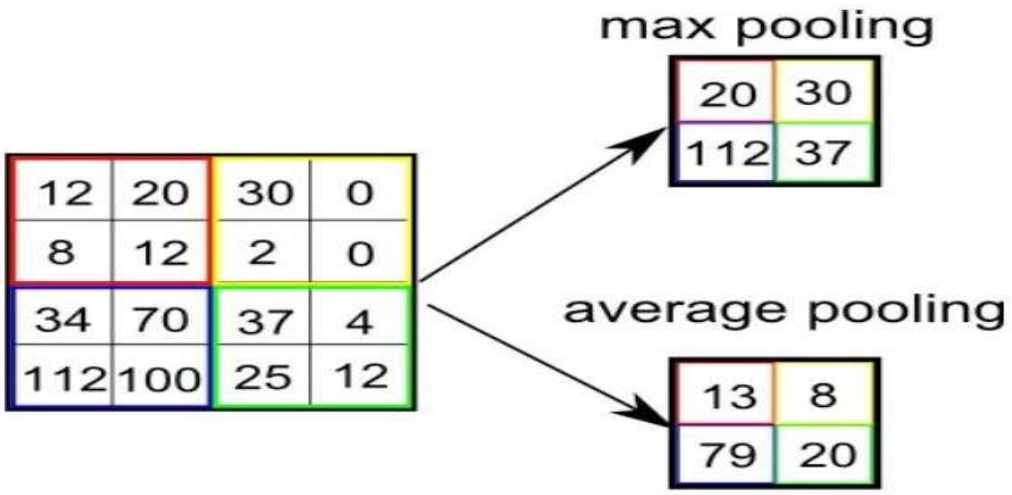

Figure 3.3. Pooling operation. From [51]

3.3 Training CNNs

The training process of CNNs is exactly the same as that of ANNs (previous chapter). First, all kernels (model parameters) are initialized according to different initialization techniques (for example, random initialization, He initialization [52], etc.). The process of learning used for adjusting these kernels is still the backpropagation algorithm, which computes the error (loss function to minimize). Then, this error is propagated back through the whole network. The gradients of the error are computed and applied using an optimization algorithm (SGD, Adam, etc.) to update the model parameters. Usually, training a CNN from scratch requires a proper initialization of parameters (weights) and can consume a lot of time. Instead, it is often recommended to initialize the parameters with a pretrained network (fine-tuning) and start the training from the point where the pre-trained network stopped its training [53]. This can help speed up the training since the model has already been trained on a large dataset and has thus learned reliable parameters.

3.4 Popular CNN architectures

In this subsection, we provide a brief description of some well-known CNN models.

LeNet (1998)

Lecun et al. [48] developed the first CNN architecture (LeNet) for automatic recognition of handwritten digits. As shown in Fig. 3.4, the LeNet architecture takes an image of size 32×32 as input (a grayscale image) and passes it through a convolutional layer C1 (with 6 kernels of size 5×5 and $sigmoid$ as the activation function) to produce 6 feature maps of size 28×28. Then, a pooling layer S2 (subsampling) is performed to reduce the size of the produced 6 feature maps to14×14. The next layer is again a convolutional layer C3 (with 16 kernels of size 5×5) that produces 16 feature maps of size 10×10. The produced 16 feature maps are again downsampled, followed by two fully connected layers of 120 and 84 units (neurons), respectively. Finally, the output layer is composed of 10 Euclidean Radial Basis Function units (RBF) corresponding to the 10 digits.

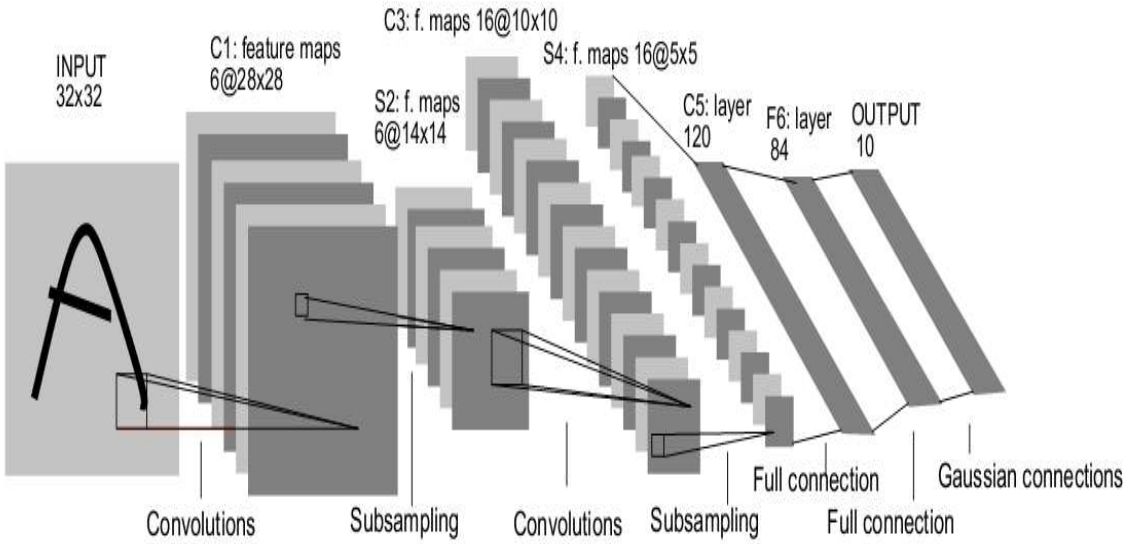

Figure 3.4. LeNet network. From [48]

AlexNet (2012)

AlexNet [54] won the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 2012 competition [55]. This competition used a subset of ImageNet database of around 1.2 million images of 1000 classes for training and 150,000 images belonging to 1000 classes for testing. AlexNet was the first CNN to realize the potential of deep learning on large datasets. As illustrated in Fig. 3.5, the input to the network is an image of size $224 \times 224 \times 3$ and the output is a $softmax$ function for classification. The network has a simple architecture, which consists of five convolutional layers that used $ReLU$ activation functions, three max-pooling layers, and three fully connected layers. The convolutional layers used kernels of different sizes (11×11 for the first, 5×5 for the second, 3×3 for the third, fourth and fifth convolutional layers). To reduce overfitting, AlexNet adopted several regularization techniques, such as Dropout and data augmentation. AlexNet was trained on two GPUs (GTX 580 GPU with 3 GB memory) for fast computation.

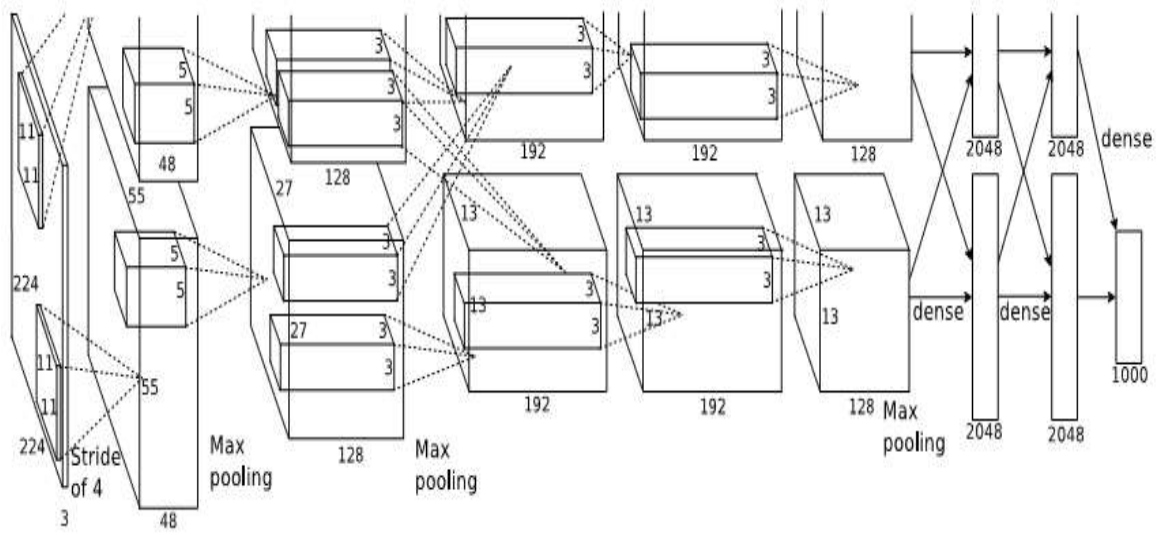

Figure 3.5. An illustration of AlexNet architecture. From [54].

VGGNet (2014)

This network won the first place in the localization task, and the second place in the classification task of the (ILSVRC) 2014 competition. It was developed by Simonyan and Zisserman [22] from the University of Oxford. VGGNet utilizes very small 3×3 filters rather than an assortment of larger filters. This work demonstrated that the stacked convolutions to increase the depth of the architecture (16 or 19 layers) by using these small filters reduces the parameter space and achieves higher performance. For example, two convolutions with filters of size 3×3 have an effective receptive field of 5×5, but the amount of parameters to estimate decreases from 25 to 18 , and three convolutions with filters of size 3×3 have an effective receptive field of 7×7, but the amount of parameters is reduced from 49 to 27. As shown in Fig. 3.6, there are 13 convolutional layers and three fully connected layers. Although the network is slow to train (because it has 138M parameters), it remains very popular. This is largely because it still performs well on image classification tasks, and has a simple structure that is easy to modify [56].

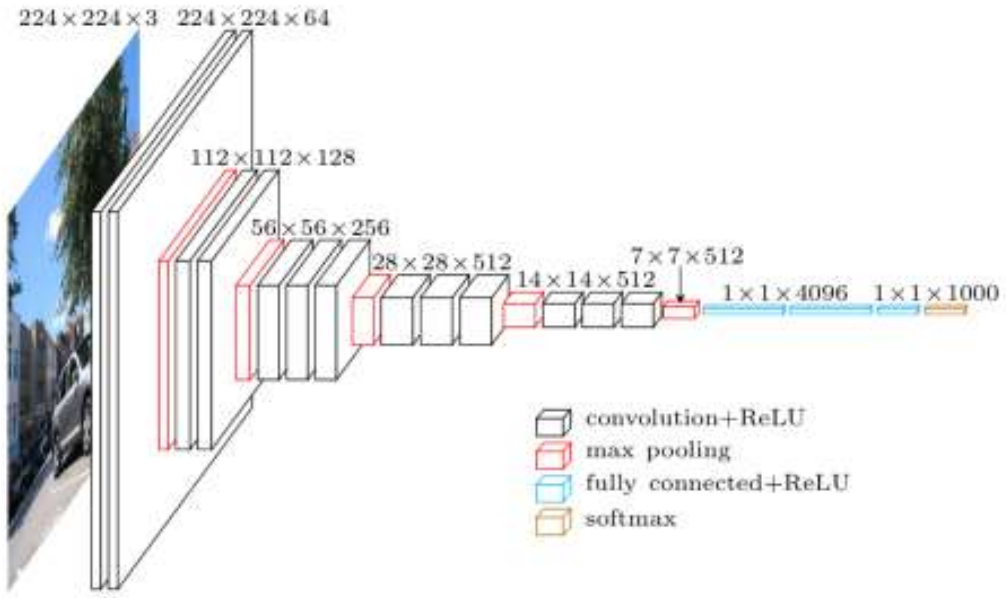

Figure 3.6. The architecture of VGG16. From [57].

GoogleNet (2015)

Szegedy et al. [58] from Google introduced GoogleNet architecture with the objective of reducing the number of parameters in the network compared to previous architectures, and capturing complex features at multi-levels. Instead of stacking convolutional and pooling layers in traditional CNNs to increase the depth of the network, the authors used new sub-networks called inception modules as multi-scale feature extractors to increase the width of the network. An inception module is shown in Fig. 3.7. It consists of 1×1, 3×3, and 5×5 convolutions and pooling layer. The outputs of these four layers are concatenated and fed into the next layer in the network. The extra 1×1 convolutions serve as bottleneck layers to reduce the dimensionality of feature maps. The network won the (ILSVRC) 2014 challenge. Other versions and extensions of inception modules are proposed in (BN-Inception [59], Inception-ResNet [60], and Xception [61]).

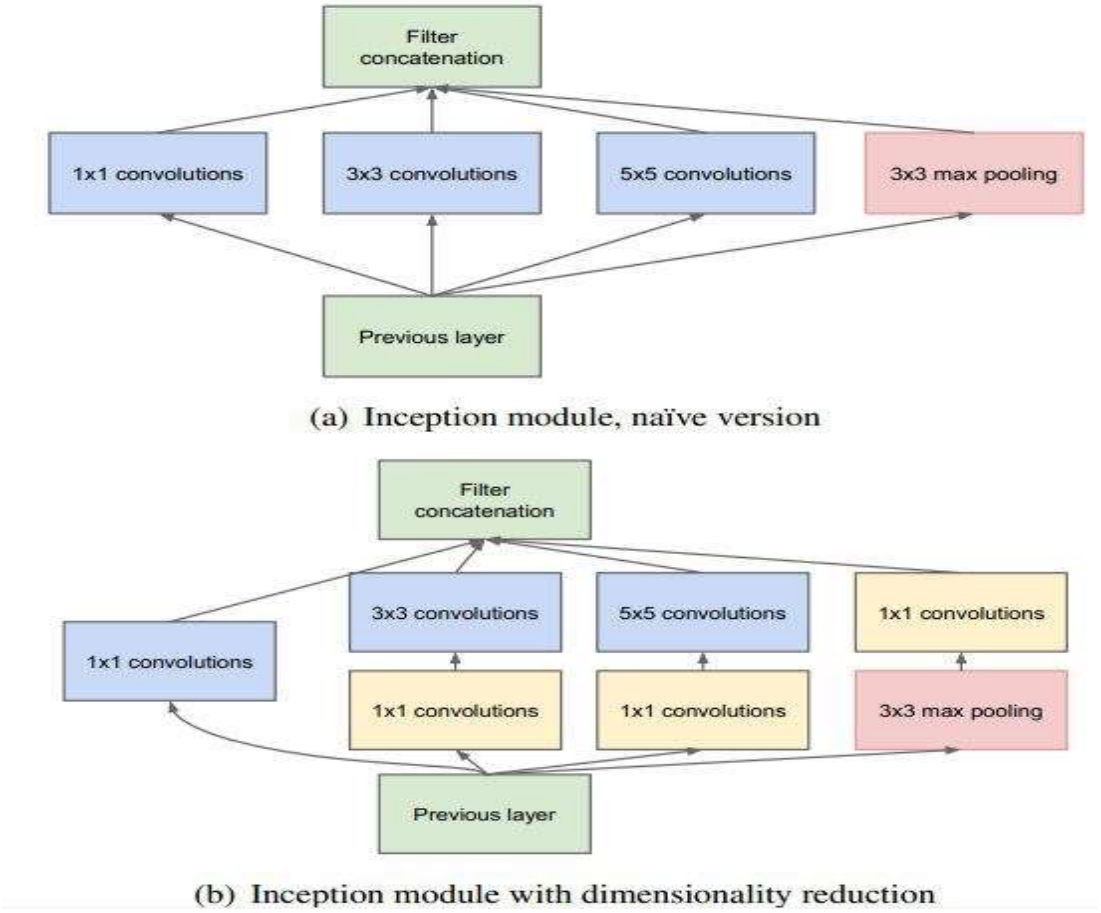

Figure 3.7. Inception module. From [58].

ResNet (2016)

He et al. [62] proposed ResNet architecture (see Fig. 3.9) that uses new building modules called residual blocks. The residual block stacks convolutional layers with residual connections (shortcut connections) as shown in Fig. 3.8. These residual connections in residual blocks alleviate the vanishing gradient problem and facilitate the training of very deep networks since a residual block learns a function with reference to the layer inputs, instead of learning unreferenced functions.

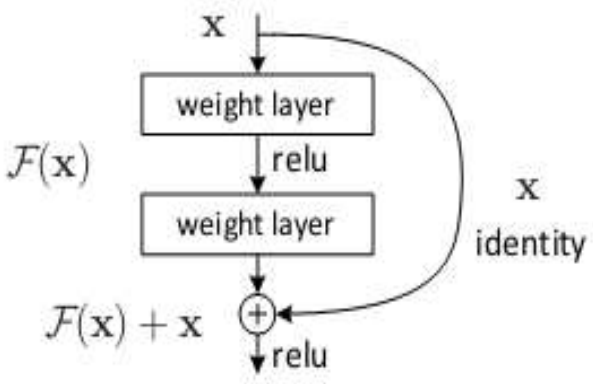

Figure 3.8. Residual block. From [62].

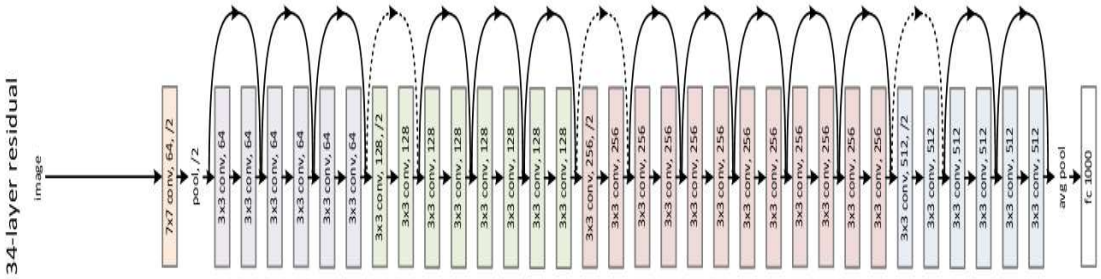

Figure 3.9. Architecture of ResNet34. From [62].

DenseNet (2017)

Huang et al. [11] proposed DenseNet, which consists of several dense blocks (Fig. 3.11). Within each dense block and via concatenation, the output feature maps of each layer are directly connected with the output feature maps of all successor layers, as illustrated in Fig. 3.10. This strategy makes the training process easier and improves the classification performance.

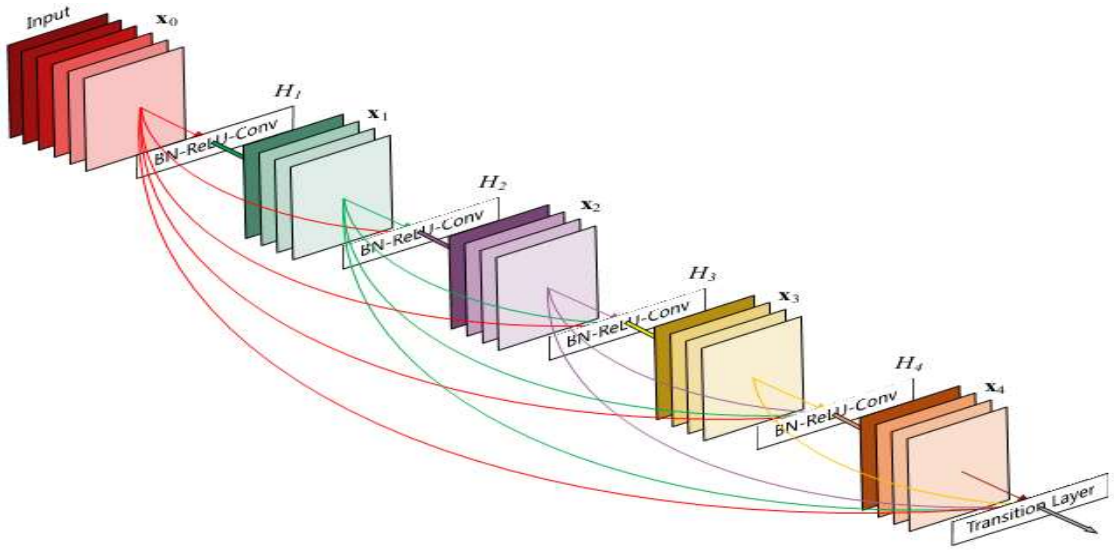

Figure 3.10. A dense block. From [11].

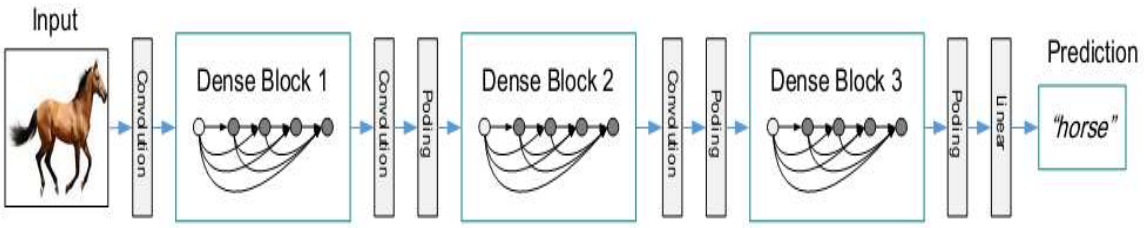

Figure 3.11. DenseNet with three dense blocks. From [11].

3.5 CNNs based semantic segmentation

In recent years, the state-of-the-art semantic image segmentation methods have been based on CNNs. This chapter presents the most popular state-of-the-art CNNs based semantic image segmentation architectures. The typical usage of a CNN is the image classification task. There are several methods how to make this network suitable for segmentation. In the following, we describe these, including sliding-window, fully convolutional networks,

encoder-decoder models, dilated convolution and DeepLab models, and pyramid pooling networks.

3.5.1 Sliding-window approach

In this approach, the segmentation can be performed by dividing the image into patches (square windows) and passing them into a CNN classifier to obtain a label for the central pixel of each patch [63]. However, this method has some drawbacks. First, it is very slow since it can only predict one pixel label by once forward computation. Secondly, there is a trade-off between computational cost and the use of context. Large patches require more computational time but allow the network to use more contextual information, whereas small patches can only use small context, but the computation cost is low.

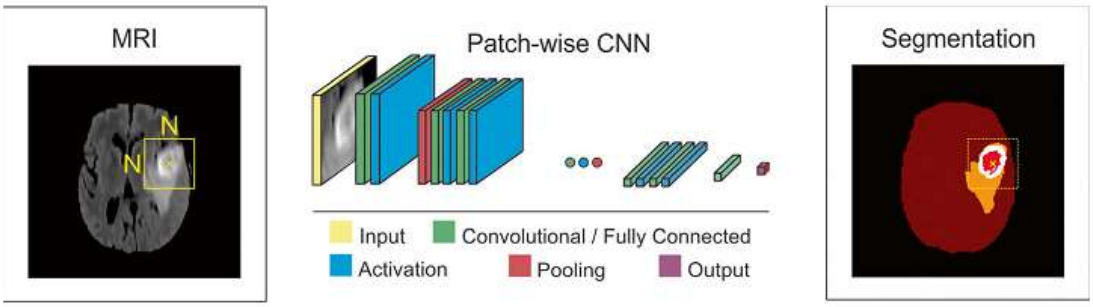

Figure 3.12. Illustration of sliding-window approach. Extracted from [64].

3.5.2 Fully convolutional networks (FCNs)

In 2015, Long et al. [23] introduced the first FCN to image segmentation task. They adapted the classification CNNs pre-trained on ImageNet (AlexNet, VGG-16, and GoogleNet) into FCNs by converting the fully connected layers to convolutional layers. The resulting fully convolutional network (FCN) can take input of any size and produce a probability map for each pixel with a single pass that is much more efficient than output for a single pixel predicted by sliding-window approach. However, because of consecutive pooling layers or striding convolutions, the resolution of the output is far lower than the input. To recover

the original input resolution, upsampling layers such as unpooling, max-unpooling, and up convolution (transpose convolution) can be used [65] (see Fig. 3.13).

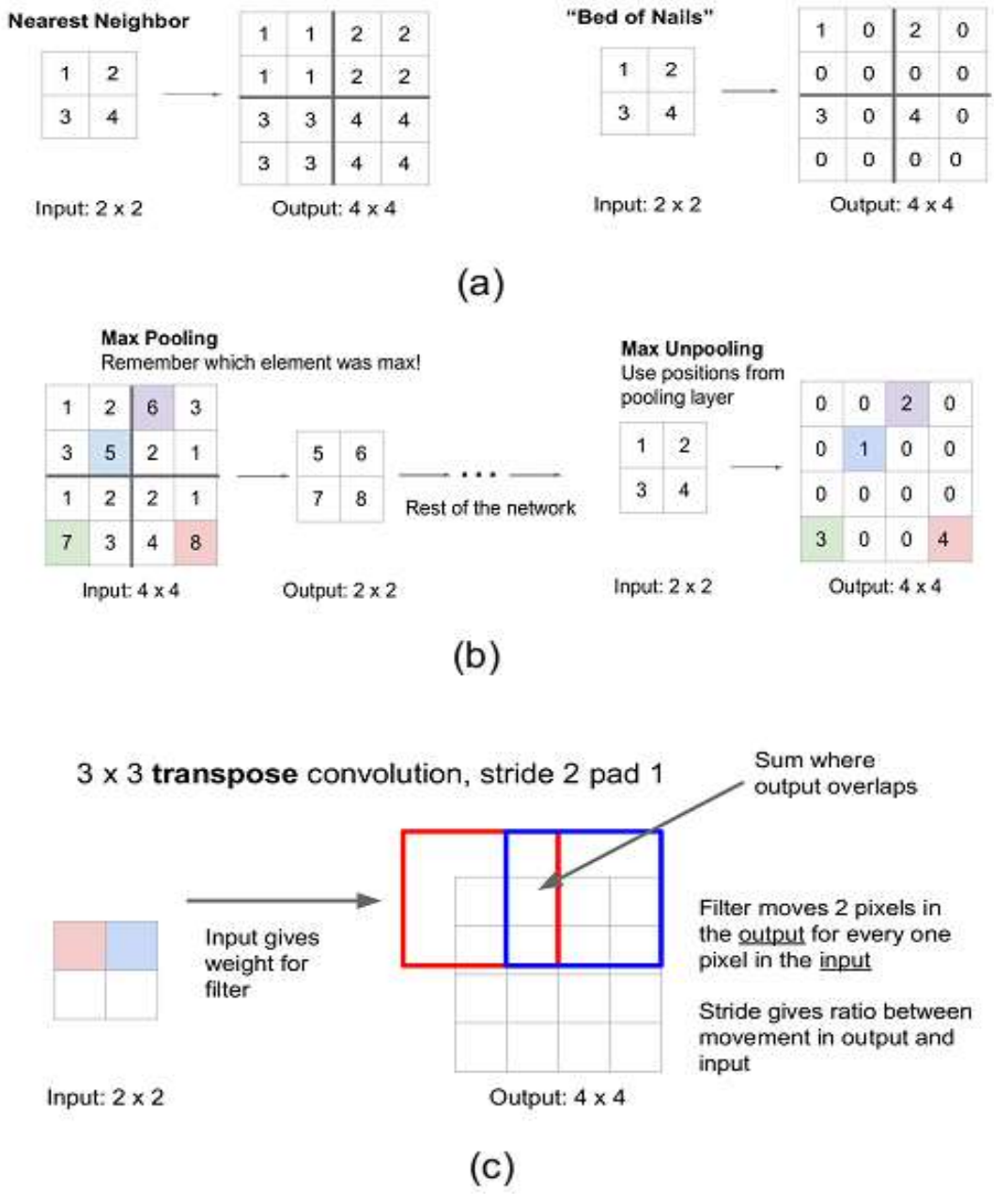

Figure 3.13. Upsampling techniques. (a) unpooling, (b) max-unpooling, (c) transpose convolution. From [65].

In FCN [23], the authors upsampled the last convolutional layer to the size of the input image (FCN32). To enhance the segmentation, the upsampled feature maps were summed with the corresponding feature maps skipped from the encoder in one (FCN16) or two (FCN8) levels (Fig 3.15).

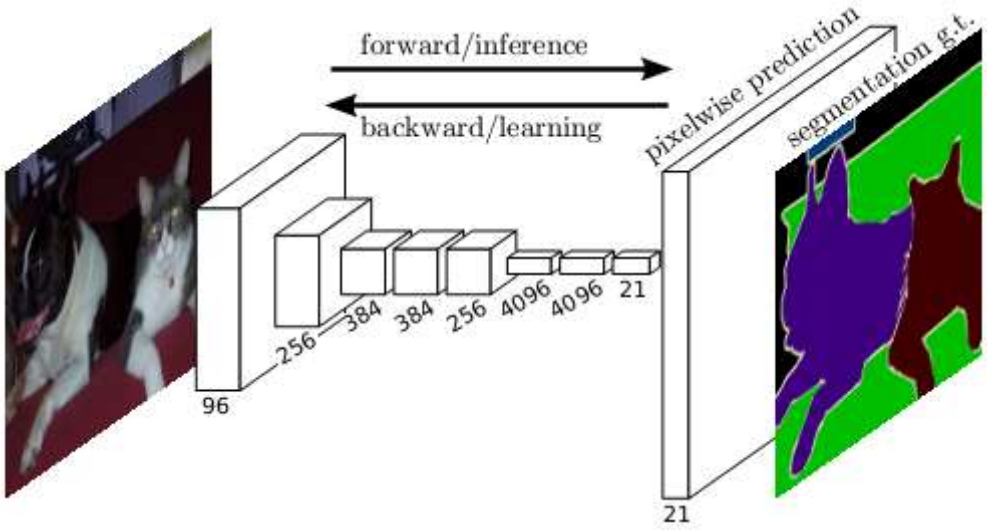

Figure 3.14. FCN. From [23].

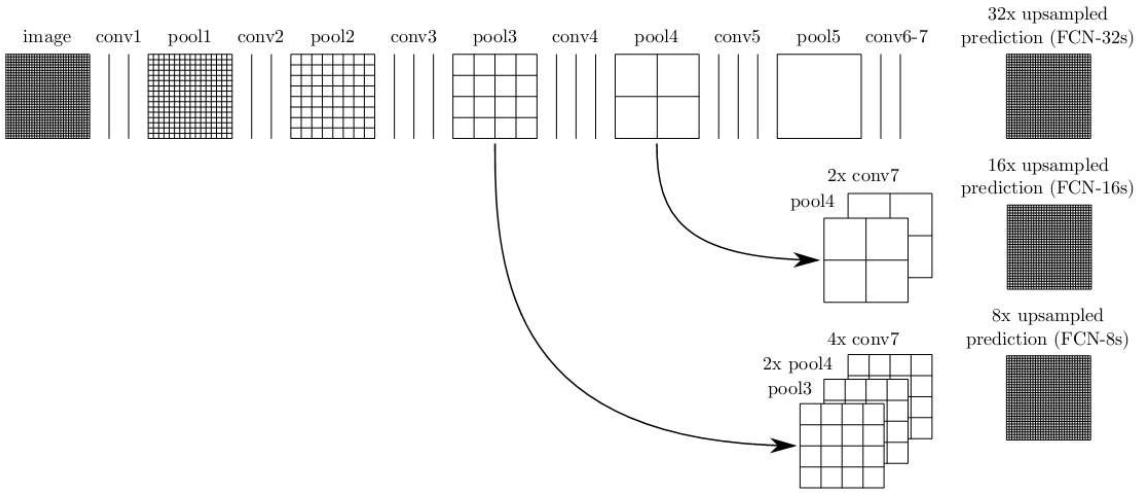

Figure 3.15. Skip connections via addition. From [23].

3.5.3 Encoder-decoder models

Most state-of-the-art CNNs adopt encoder-decoder architectures, such as DeconvNet [66], SegNet [24], U-Net [14], and FC-DenseNet [67].

DeconvNet

Noh et al. [66] proposed a deep encoder-decoder network (DeconvNet). The encoder part corresponds to a feature extractor that has the same topology as VGG16 [22] excluding the last classification layer. The contribution of DeconvNet lies in the decoder part (deep deconvolution network) that takes as input the feature representation and generates a map of pixel-wise probabilities. Unlike FCN, the decoder part is composed of multiple series of unpooling and deconvolution layers to construct dense pixel-wise class prediction map.

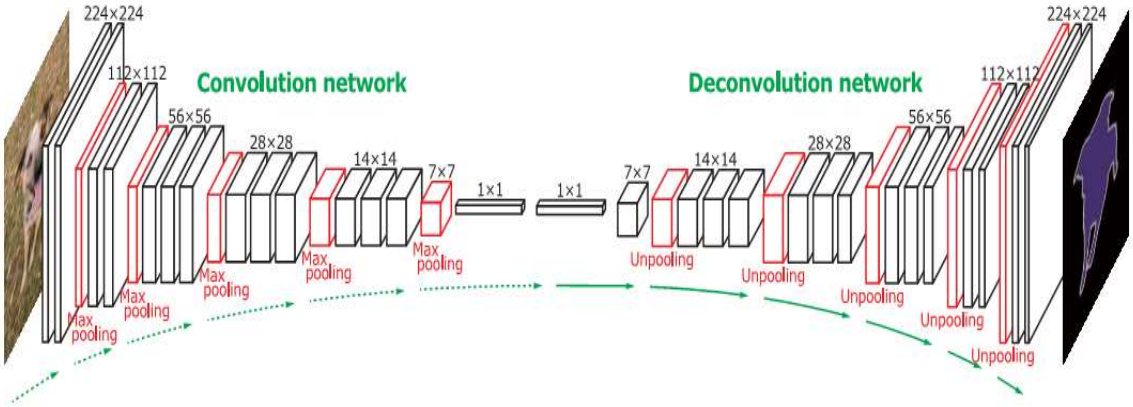

Figure 3.16. DeconvNet architecture. From [66]

SegNet

Badrinarayanan et al. [24] developed a symmetric encoder-decoder segmentation network called SegNet (see Fig. 3.17). Similar to DeconvNet, the encoder part has the same topology as VGG16, but without any fully connected layers, which makes the encoder part smaller and easier to train than DeconvNet. The SegNet encoder consists of 13 convolutional layers that extract deep feature maps, and the same layers are mirrored in the

decoder part. The decoder part consists of convolution and upsampling layers that use pooling indices computed in the max-pooling from the corresponding encoder part.

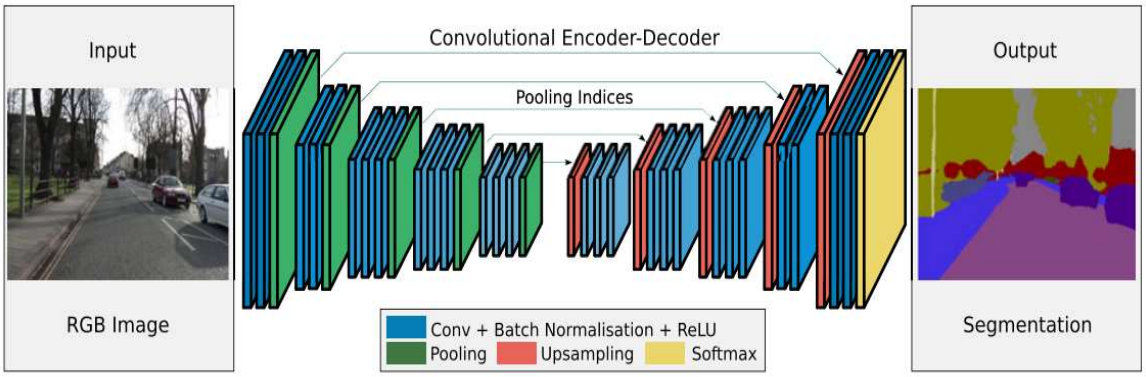

Figure 3.17. SegNet. From [24].

U-Net

U-Net [14] is the most well-known and commonly used in medical image segmentation. Similar to FCN, DeconvNet, and SegNet, U-Net contains two components: an encoding path (encoder) and a corresponding decoding path (decoder). The encoding path is composed of convolutional and pooling layers to extract high-level semantic information (deep features), and the decoding path is composed of transposed convolutional and convolutional layers to perform segmentation (dense prediction). To help recover the missing detail lost by pooling layers, U-Net concatenates the low-level feature maps of the encoding path with the corresponding feature maps of the decoding path. Several extensions of U-Net have been developed in nearly for all imaging modalities [68]. For example, Zhou et al. [25] proposed a nested U-Net architecture (U-Net ++) where the encoder and the decoder sub-networks are connected through a series of nested, dense skip pathways. They conducted experiments on four medical imaging datasets of different modes. Ibtehaz et al. [69] proposed MultiResUNet model that replaces the sequence of two convolutional layers and skip connections with the proposed MultiRes block and Res paths, respectively. MultiResUNet has also been applied to a repertoire of multimodal medical images.

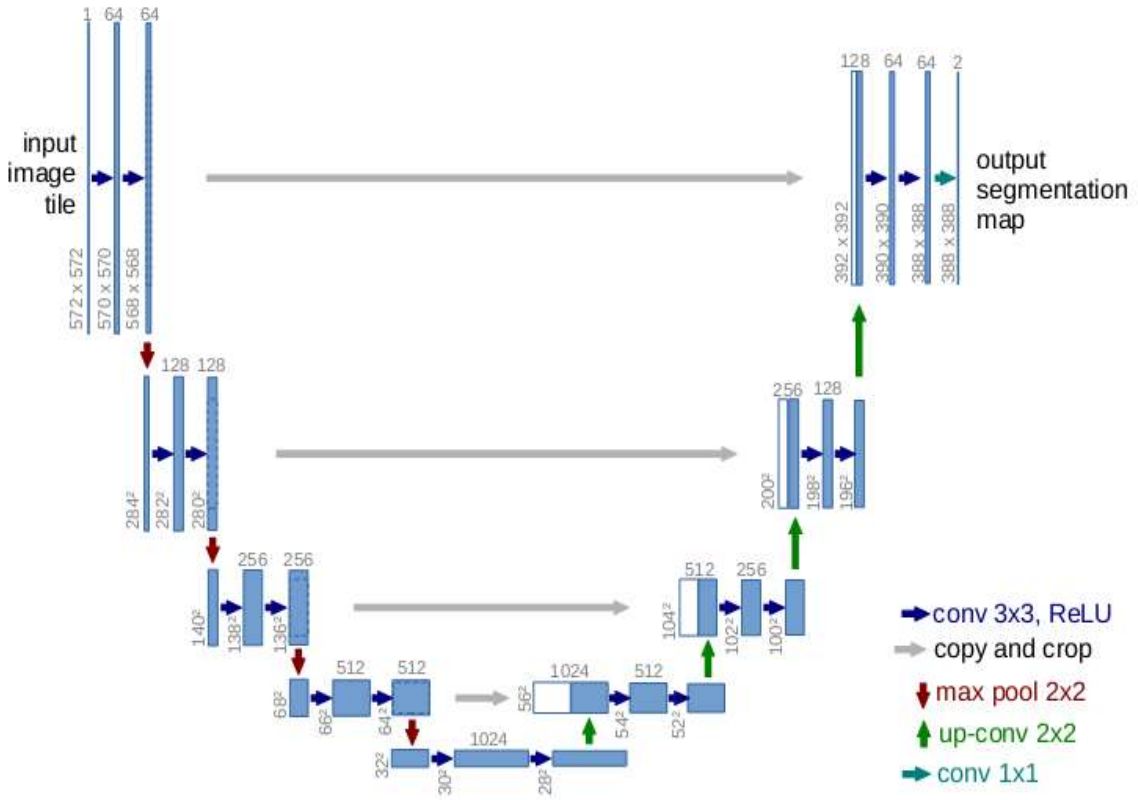

Figure 3.18. Architecture of U-Net. From [14].

FC-DenseNet

Jégou et al. [67] extended the classification network DenseNet to a fully convolutional network for semantic segmentation (FC-DenseNet). As illustrated in Fig. 3.19. The downsampling path (encoder) comprises convolution, dense blocks, and Transitions Down (TD). On the other hand, the upsampling path (decoder) includes Transitions Up (TU), skip connections, and convolution. Note that in the encoder, the input to a dense block is concatenated with its output, except for the last one, which is used as a bottleneck.

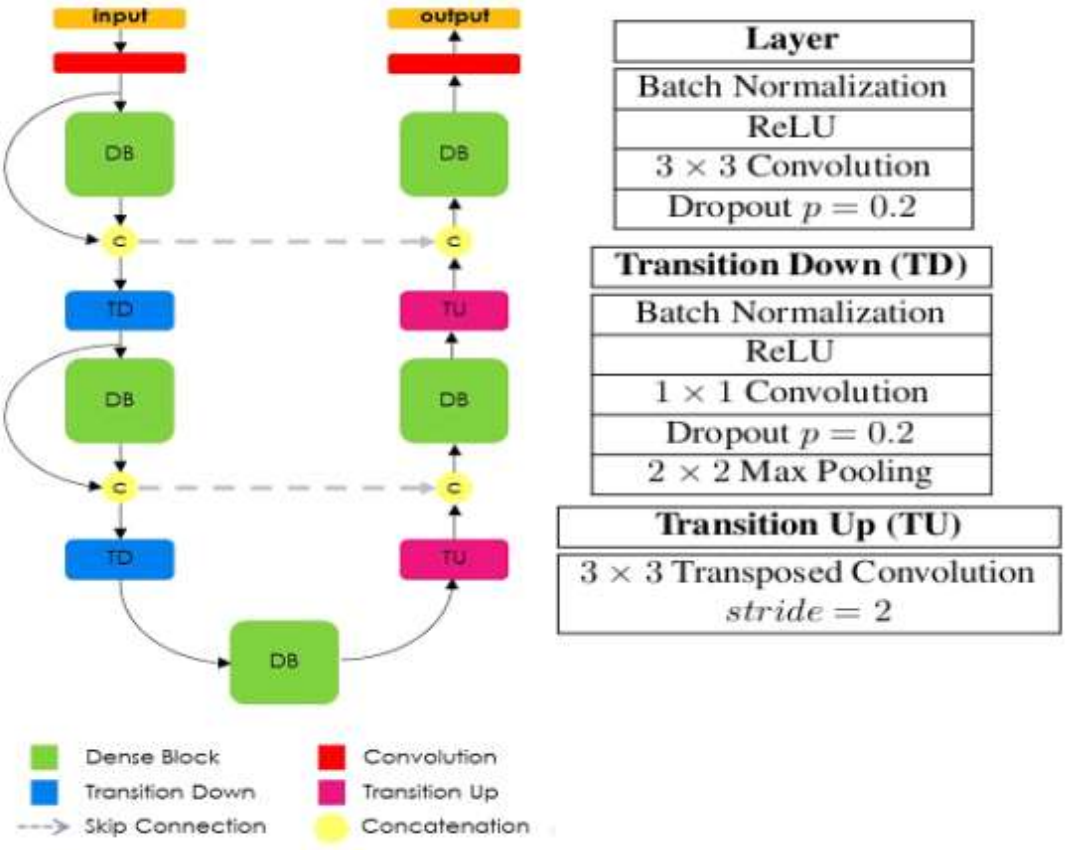

Figure 3.19. FC-DenseNet architecture. From [67]

3. 5.4 Dilated convolution and DeepLab models

Dilated convolution

Instead of downsampling feature maps, Yu and Koltun [16] developed a convolutional network module using dilated convolutions to aggregate multiscale contextual information for dense prediction. Dilated convolutions enlarge the receptive field of the network without reducing spatial resolution by inserting "zeros" in the convolution kernels (Fig. 3.20). Fig. 3.21 summarizes the proposed context modules (basic and large), which were plugged into the front-end network. The latter is a truncated VGG16 obtained by removing the last two pooling and striding layers.

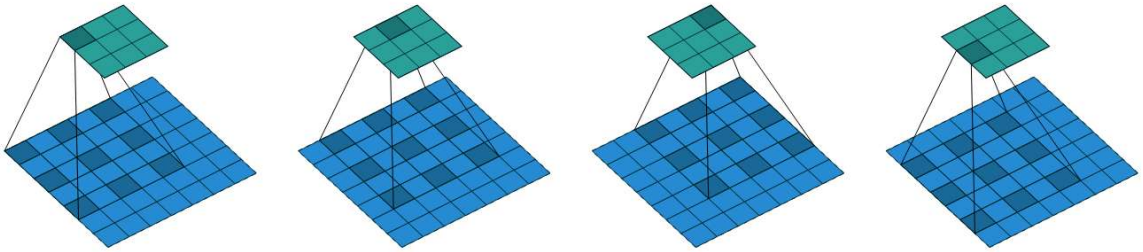

Fig 3.20. Dilated convolution with dilation of 2. From [70].

Layer	1	2	3	4	5	6	7	8
Convolution	3×3	3×3	3×3	3×3	3×3	3×3	3×3	1×1
Dilation	1	1	2	4	8	16	1	1
Truncation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Receptive field	3×3	5×5	9×9	17×17	33×33	65×65	67×67	67×67
Output channels								
Basic	C	C	C	C	C	C	C	C
Large	$2C$	$2C$	$4C$	$8C$	$16C$	$32C$	$32C$	C

Fig 3.21. Convolutional network module. From [16]. By using different dilations, information in multiple scales can be sampled and then concatenated.

Since the use of dilated convolution can cause gridding artifacts, Yu et al. [71] developed a scheme that uses residual connections and called dilated residual networks (DRN) to alleviate these artifacts and further increase the performance.

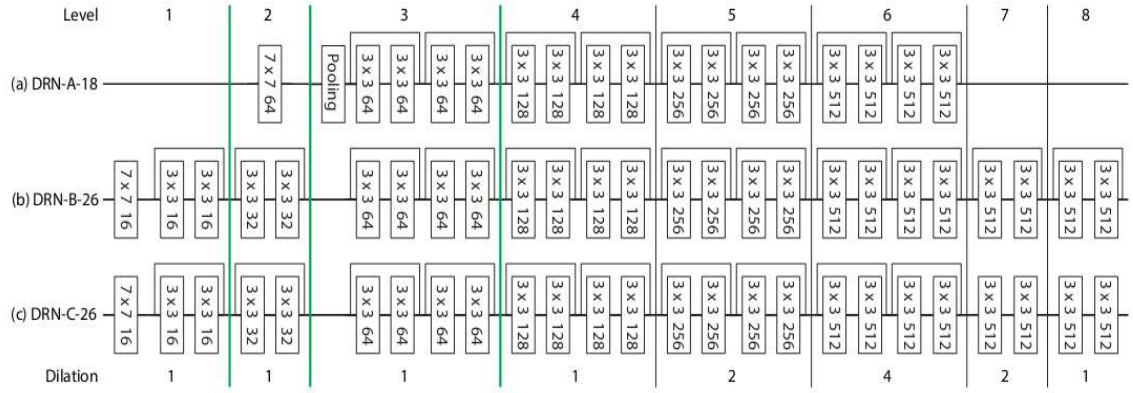

Figure 3.22. DRN architectures. From [71]. The bold green lines represent the down-sampling. The output feature maps were upsampled to full resolution using bilinear interpolation.

DeepLab models

Chen et al. [18] presented a DCNN called DeepLab for semantic segmentation. In their network, they replaced the last two max-pooling operations from the VGG16 network by dilated (atrous) convolutions to maintain the feature resolution unchanged to incorporate larger context. To improve the localization of object boundaries, they appended their pipeline with the fully connected conditional random field (CRF) (see Fig. 3.23).

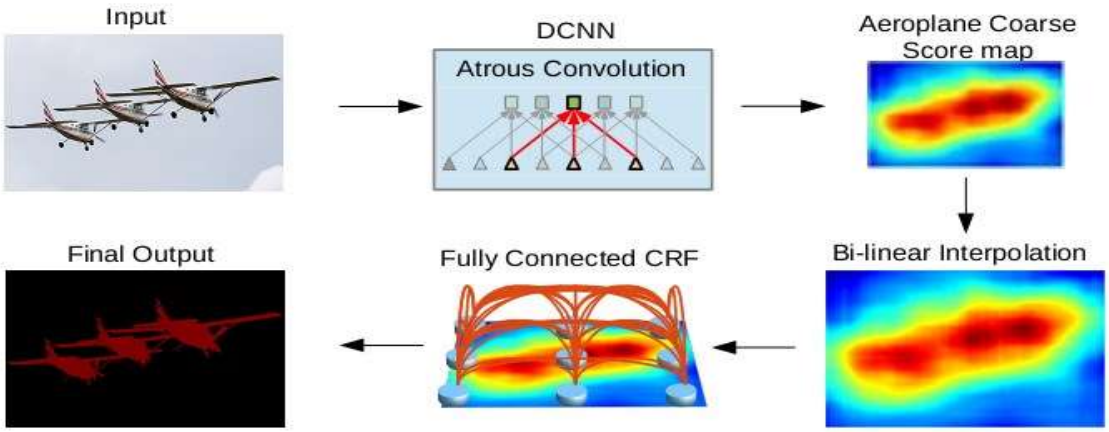

Fig 3.23. DeepLab model illustration. From [18]

An improvement of DeepLab, which is DeepLabv2, introduced the atrous spatial pyramid pooling module (ASPP), where parallel dilated convolutions with different dilation rates were performed to segment objects at multiple scales.

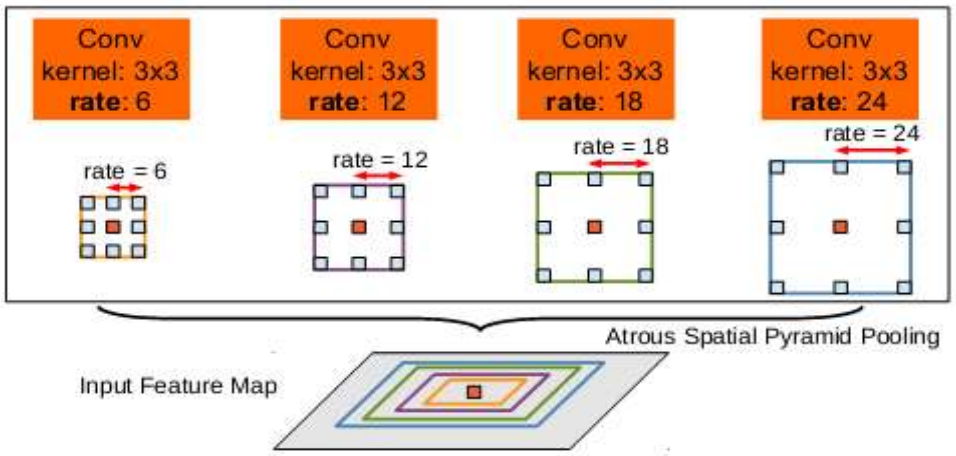

Figure 3.24. Atrous Spatial Pyramid Pooling module (ASPP). From [18].

Based on the DeepLabv2 network, DeepLabv3 [19] revisited atrous convolution and designed modules that employ atrous convolution in cascade or in parallel, and included batch normalization within ASPP. Furthermore, global average pooling was added to encode the global context to further boost the segmentation performance.

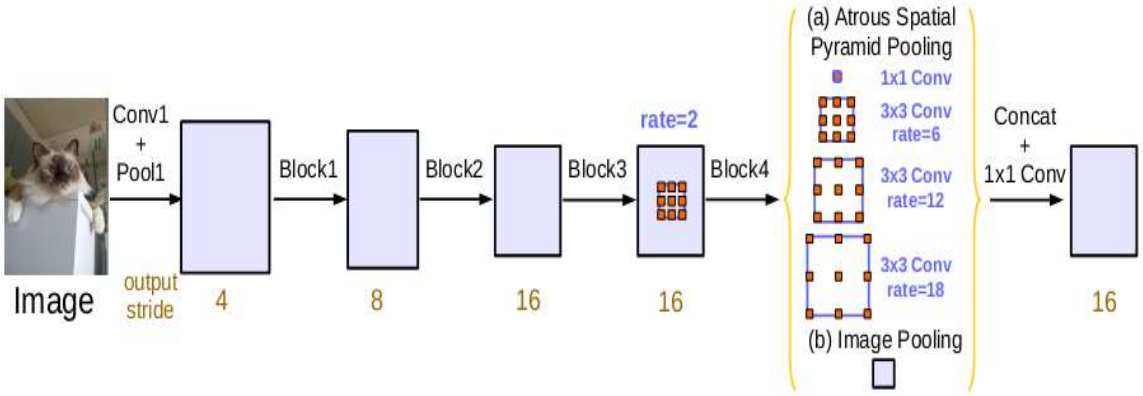

Figure 3.25. Improved ASPP. It consist of 1×1 convolution, three 3×3 convolutions with differents dilation rates, all with batch normalization, and image-level features (global average pooling). From [19].

Lately, to refine the object boundaries, Chen et al [20] added a decoder path to DeepLabv3 and called it DeepLabv3+. In order to build a faster and stronger network, they employed the Xception model [61] as the network backbone and introduced depthwise separable convolution into ASPP and decoder modules.

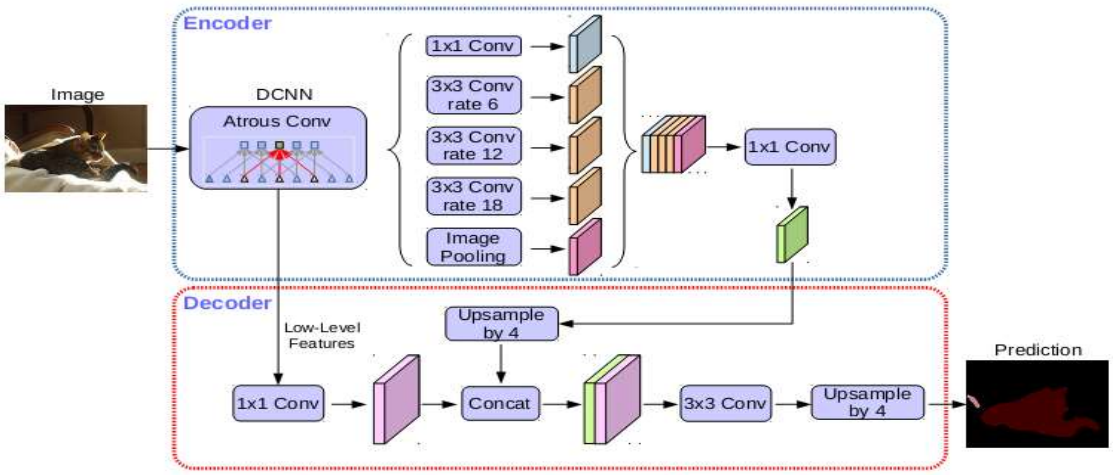

Figure 3.26. DeepLabv3+. From [20].

3.5.5 PSPNet

PSPNet [17] provides an effective global prior representation for pixel-wise scene parsing. As shown in Fig. 3.27, the authors proposed a pyramid pooling module (PPM) at the end of the backbone network. This PPM used four pooling layers with different kernel sizes: 1×1, 2×2, 3×3, and 6×6. As a result, levels of information (feature maps) were collected. To reduce the number of feature maps, 1×1 convolutional layer was used after each pyramid level. Then, these low-dimension feature maps were upsampled (via bilinear interpolation) to the original sizes and concatenated with the original input features. Finally, this final feature representation was fed into convolutional layer to get the pixel-wise prediction.

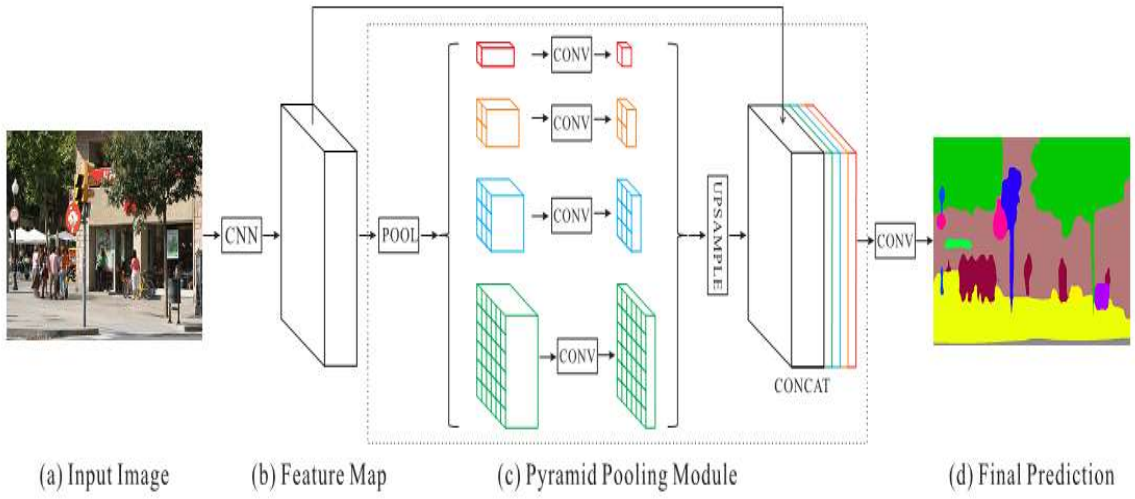

Figure 2.27. Overview of PSPNet. From [17].

3.6 Conclusion

This chapter focused on CNNs, which are the most commonly used in computer vision and medical image analysis, and achieved the state-of-the-art performance. We presented some popular CNN based models for semantic image segmentation. Other CNN based models and a comprehensive review of deep learning based architectures for semantic segmentation of natural and medical images can be found in several surveys [72-80]. The

success of CNN encouraged us to orient this thesis to tackle the problem of skin lesion segmentation using this network as the backbone of the proposed approaches.

Chapter 4

Skin lesion segmentation methods for dermoscopy images

4.1 Introduction

In recent years, dermoscopy has been widely used for melanoma detection. Dermoscopy (also known as epiluminescence light microscopy, dermatoscopy, or skin surface microscopy) is a non-invasive imaging technique that uses optical magnification, liquid immersion, and a light source or cross-polarization of a light source to make the contact area (epidermis) translucent, consequently increasing the visibility of subsurface structures of the skin [10, 81]. Various methods have been developed for melanoma detection, such as the ABCD rule (Asymmetry, Border, Color, and Differential structure) [82], the Menzies method [83], the 7-point checklist [84], the CASH algorithm (Color, Architecture, Symmetry, and Homogeneity) [85], and pattern analysis [86]. However, manual analysis of dermoscopy images is time-consuming, subjective, and poorly reproducible. In this regard, computer aided diagnosis systems (CADS) can be used to enhance the diagnosis performance. One of the most important steps in CAD systems is the automated and accurate segmentation of skin lesions. This task is a very challenging issue due to the various factors, such as the presence of hair, blood vessels, and bubbles; some lesions have irregular and fuzzy boundaries, very low contrast between the lesion and the surrounding healthy skin, wide variation regarding sizes and colors.

Figure 4.1. A dermoscope. From [87]

The rest of this chapter is organized as follows: the next section provides an overview of skin anatomy and diseases. Then, we present the datasets and metrics used for skin lesion segmentation in section 4.3 and section 4.4, respectively. The skin lesion segmentation methods, including unsupervised methods, traditional supervised methods, and especially

deep learning-based methods are described in section 4.5. Finally, the section 4.6 concludes the chapter.

4.2 Skin Anatomy and diseases

Skin is the largest organ in the human body in terms of both surface and weight. It has a surface area of $2m^2$ and a weight of 3.6 kg for an adult [88]. It mainly acts as a barrier to the exterior environment. It protects the deeper body tissues from injury, helps regulate body temperature, protects the body from harmful effects of UV radiation, synthesizes vitamin D, and provides sensory information (pressure, pain, heat, and cold). The skin consists of three layers of tissue (Fig. 4.2): the epidermis, the dermis, and the hypodermis (subcutis), all with different thicknesses and functions. The epidermis is the outermost layer of skin and is made up of five sub-layers: stratum corneum (the top layer), stratum lucidum, stratum granulosum, stratum spinosum and stratum basale (the base layer). The epidermis contains melanocytes. These cells produce melanin, which gives the skin its color to protect against UV radiation. In addition to melanocytes, the epidermis also contains other specialized cells, such as keratinocytes (the majority of cells in it) that produce keratin, Langerhans cells that have an immunologic function, and sensory recipient cells that called Merkel cells. The dermis is the middle layer (under the epidermis). It contains different types of cells, such as collagen fibers, blood vessels, and fibroblasts. It gives the skin toughness, strength, and flexibility and nourishes the epidermis, which has no blood vessels [89]. The third layer is the hypodermis, which is a fatty layer. It functions as a reserve source of energy and forms the link between the skin and the rest of the body [89]. Skin disease is one of the most common diseases that occur all over the world. There are various types of skin diseases according to their origin and degree of malignancy [90] (Fig. 4.3). The common malignant skin diseases are basal cell carcinoma (BCC), squamous cell carcinoma (SCC), and the most deadly type called melanoma. The main cause of these cancers is due to increased ultraviolet (UV) exposure.

Basal cell carcinoma

It is the most common but the least malignant type of skin cancer. It starts in the stratum basale (basal cells), which is the deepest layer of the epidermis. This cancer usually develops on areas of skin that are exposed to the sun, such as the face, head, and neck.

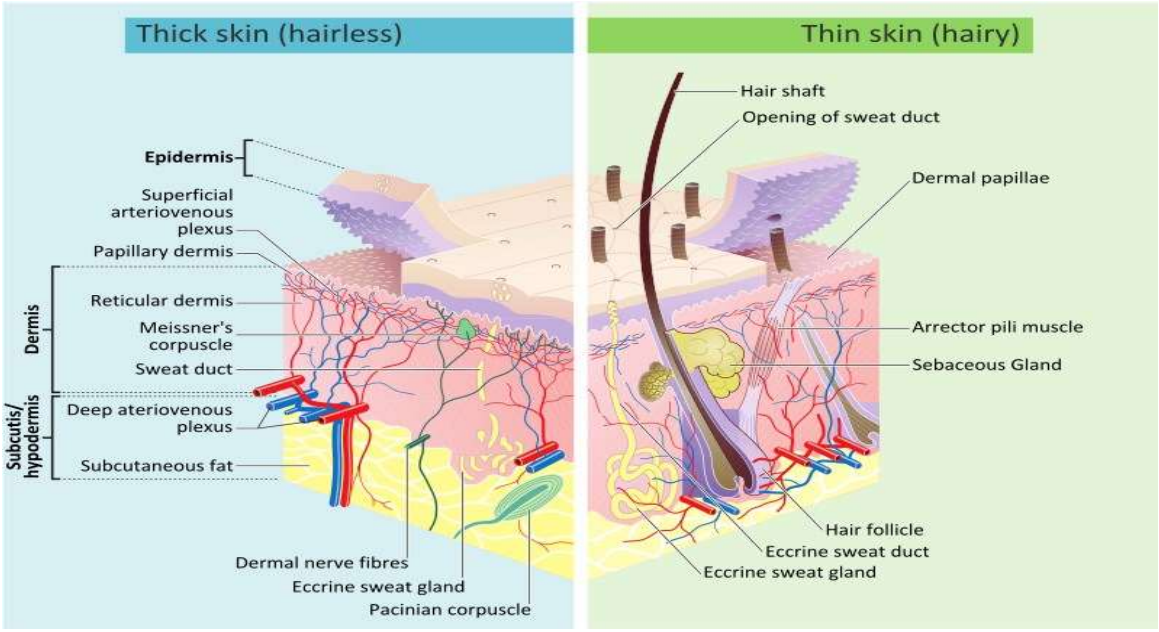

Figure 4.2. Structure of the skin. Image from [91]

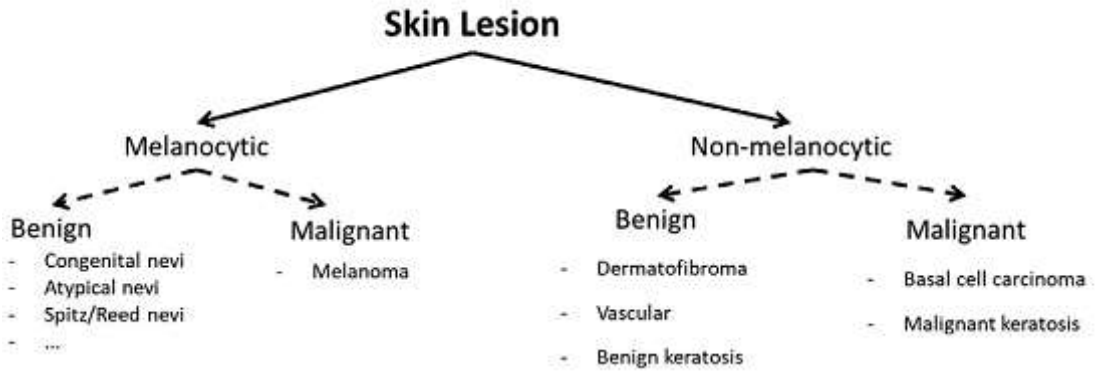

Figure 4.3. Skin diseases types. From [90]

BCCs grow slowly and rarely spread to other parts of the body. But if allowed to grow, it can be dangerous and invade the bone or other tissues beneath the skin [92][93].

Squamous cell carcinoma

This cancer is the second most common type of skin cancer. It starts in the squamous cells located above the stratum basale. SCCs commonly occur on sun-exposed areas of the body. Untreated SCCs can become invasive, and spread to other parts of the body [94].

Malignant melanoma

This cancer is the least common, but the deadliest type of skin cancer. Melanoma arises when melanocytes (the cells in the epidermis that produce melanin) begin to grow out of control. It can develop on areas of the body that are never exposed to the sun. Melanoma is characterized by its high metastasis rate, and thus the treatment would be complicated and deadly [95]. Therefore, early detection of melanoma is essential because it can increase the survival rate of patients.

4.3 Dermoscopic lesion segmentation datasets

In this subsection, we present the common publicly available datasets for skin lesion segmentation from dermoscopic images. For a more detailed overview of other datasets used for skin disease diagnosis, including three types of modalities (dermoscopic images, clinical images, and pathological images), the readers may refer to [50, 95, 96].

PH2 dataset

The PH2 dataset [97] is provided by the Universidade do Porto, Tecnico Lisboa, and the Dermatology Service of Hospital Pedro Hispano in Portugal. This dataset contains 200 dermoscopy images and their ground truths (160 non-melanoma and 40 melanoma), and were acquired under the same conditions through Tuebinger Mole Analyzer system using using a magnification of $20\times$. All images are 8-bit RGB in BMP format with size varies from 553×763 pixels to 577×769 pixels.

ISIC Dataset

In recent years, the well-established public benchmark datasets used in the literature have been provided by the International Skin Imaging Collaboration (ISIC) archive [98], which

contains dermoscopic images collected from a variety of different leading international clinical centers, acquired from various devices used at each center. ISIC organizes yearly a challenge named skin lesion analysis toward melanoma detection at the IEEE International Symposium on Biomedical Imaging (ISBI). The goal of the challenge is to provide a dataset of dermoscopic skin images to boost the performance of melanoma diagnosis. For the segmentation task, three datasets were provided from 2016 to 2019 as follows.

- ISBI 2016 dataset [99] contains 900 training annotated images (727 non-melanoma and 173 melanoma). For evaluation, another set of 379 images (304 non-melanoma and 75 melanoma) and their ground truths were provided. These images are 8-bit RGB in JPG format with size varies from 566×679 pixels to 2848×4228 pixels.
- The ISBI 2017 dataset [100] contains 2000 training dermoscopy images and their ground truths (1626 non-melanoma and 374 melanoma), 150 annotated images (120 non-melanoma and 30 melanoma) for validation, and another set of 600 annotated images (483 non-melanoma and 117 melanoma) for evaluation (testing). All images are 8-bit RGB in JPG format with different sizes from 540×722 pixels to 4499×6748 pixels; while the ground truths are binary masks in PNG format.
- The ISBI 2018 dataset [101, 102] consists of 3694 RGB dermoscopic images with 2594 training annotated images, 100 images in the validation set and 1000 in the testing set; and both of them without corresponding segmentation masks. Image size varies from 576×768 to 6748×4499.

The distribution of these datasets is summarized in Table 4.1, while Fig. 4.4 illustrates some images in the PH2 and ISBI 2017 datasets.

Dataset	Lesion type	Training data	Validation data	Test data
PH2 [97]	Common Nevi	-	-	80
	Atypical Nevi	-	-	80
	Melanoma	-	-	40
ISBI 2016 [99]	Melanoma	173	-	75
	Non-melanoma	727	-	304
ISBI 2017 [100]	Melanoma	374	30	117
	Seborrheic Keratosis	254	42	90
	Benign Nevi	1372	78	393
ISBI 2018 [101, 102]	-	2594	100	1000

Table 4.1. The distribution of PH2, ISBI2016, ISBI2017, and ISBI 2018 datasets.

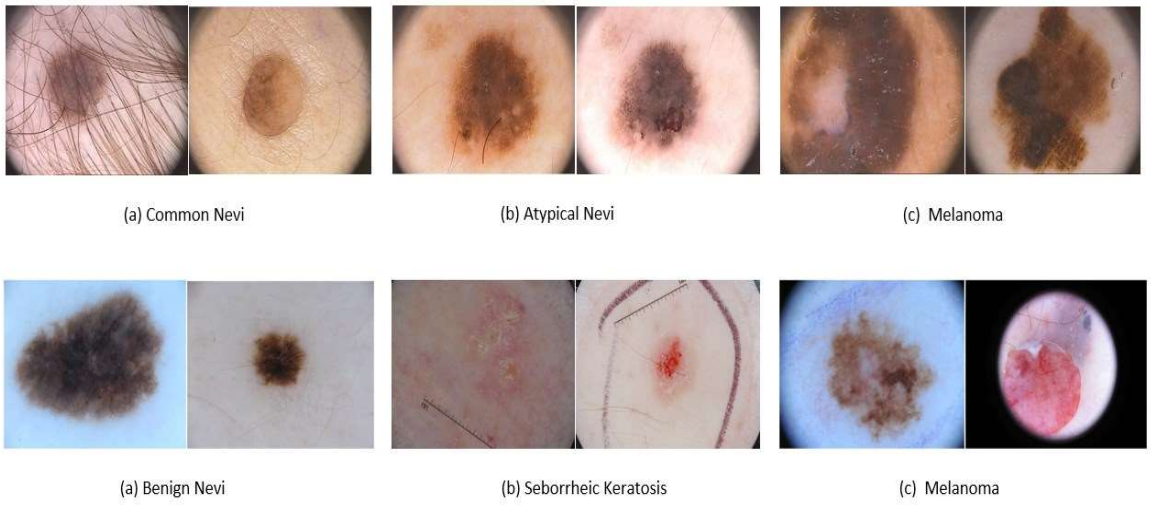

Figure 4.4. Examples of skin lesion images in PH2 (the first row), ISBI 2017 (the second row).

4.4 Performance evaluation metrics

The performance of the segmentation algorithms is assessed by several metrics. The commonly used are: Jaccard index (JAC), dice coefficient (DIC), sensitivity (SEN), specificity (SPE), and accuracy (ACC). As a result of automatic segmentation, there are four possible outputs, which represent the elements of the confusion matrix (Fig. 4.5) and are used to calculate the aforementioned metrics:

True positives (TP): the number of lesion pixels that were correctly segmented. True negatives (TN): the number of non-lesion pixels (background, healthy skin) that were correctly segmented.

False positives (FP): the number of non-lesion pixels that were incorrectly segmented. False negatives (FN): the number of lesion pixels that were incorrectly segmented.

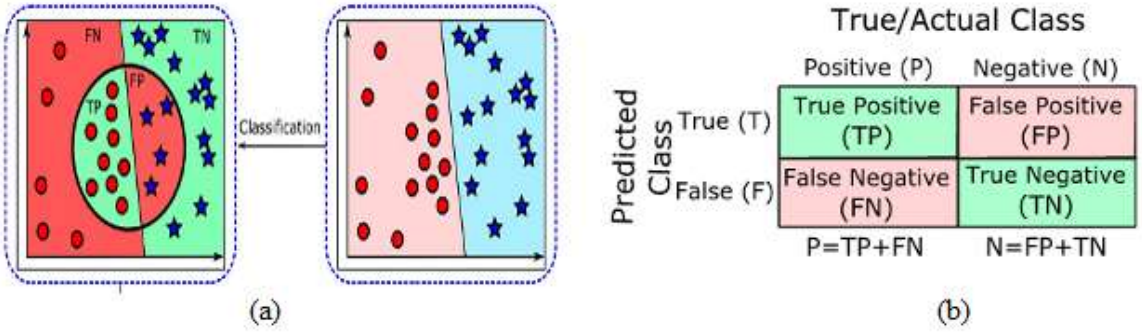

Figure 4.5. Confusion matrix (b) contains the output results given by the classifier (a) (the black circle) on a dataset with two classes. From [103].

Jaccard index (JAC) and dice coefficient (DIC) are the most common evaluation metrics. They measure the similarity between the predicted segmentation mask (PR) and the ground truth (GR), and are defined as:

$$JAC = \frac{|PR \cap GR|}{|PR| + |GR| - |PR \cap GR|} \tag{4.1}$$

$$DIC = \frac{2*|PR \cap GR|}{|PR|+|GR|} \quad (4.2)$$

At pixel level, they can be also calculated as:

$$JAC = \frac{TP}{TP+FP+FN} \quad (4.3)$$

$$DIC = \frac{2.TP}{2.TP+FP+FN} \quad (4.4)$$

Sensitivity (SEN) : this is the proportion of lesion pixels that were correctly classified. It is defined as:

$$SEN = \frac{TP}{TP+FN} \quad (4.5)$$

Specificity (SPE) : this is the proportion of non-lesion pixels (healthy skin) that were correctly classified. It is given as:

$$SPE = \frac{TN}{TN+FP} \quad (4.6)$$

Accuracy (ACC) : this is the proportion of pixels that were correctly classified (both TP and TN). It is expressed as:

$$ACC = \frac{TP+TN}{TP+FP+FN+TN} \quad (4.7)$$

4.5 Skin lesion segmentation methods

To address the problem of automatic skin lesion segmentation, various methods have been proposed, and they can be categorized into three groups [10]: unsupervised methods, traditional supervised methods, and deep learning-based methods.

4.5.1 Unsupervised methods

These techniques do not require training data and generally fall into one of the following groups.

Thresholding techniques

These techniques have been widely used in literature. They use the histogram of the input image to separate the lesion from the background (skin) by the determination of one or more threshold values [7, 104]. Celebi et al. [104] proposed an automated method for detecting lesion borders in dermoscopy images. In their work, they fused the results provided by an ensemble of thresholding methods such as Huang threshold [105], Kapur threshold [106], Kittler threshold [107], and Otsu threshold [108]. Yuksel and Borlu [109] introduced a method that uses type-2 fuzzy logic techniques [110] to automatically determine the threshold value for accurate segmentation of dermoscopic images.

Clustering techniques

They attempt to partition the color space of dermoscopy images into homogeneous regions [111]. Zhou et al. [112] presented a new mean shift based fuzzy c-means method that incorporates a mean field term within the standard fuzzy c-means objective function. The proposed segmentation method required less computational time than other fuzzy c-means (FCM) algorithms. Furthermore, it provided superior performance. Castillejos et al. [113] proposed a novel approach for dermoscopic image segmentation based on wavelet transform (WT) for k-means, fuzzy c-means (FCM), and cluster preselection fuzzy c-means (CPSFCM) techniques.

Edge-based methods

They utilize edge operators. An edge in image processing is an area with abrupt changes in the intensity value (gray level). In edge-based methods, segmentation is done by identifying the discontinuities to detect boundaries in the image. Abbas et al. [114] presented an automated method to detect lesion borders in dermoscopy images. Their method started with a pre-processing step where artifacts were removed using homomorphic transform filtering [115], a weighted median filter and an exemplar-based object removal algorithm [116]. Then a least-squares method (LSM) [117] was performed to acquire edge points. Finally, the dynamic programming (DP) technique [118, 119] was used to find the optimal lesion border.

Region-based methods

These techniques utilize region splitting, which partitions the image into several sub-regions having similar characteristics (the neighboring pixels within one sub-region are similar, with respect to a set of pre-defined criteria), region merging, or both [7, 120]. Celebi et al. [120] introduced a technique to detect lesion borders in dermoscopy images using the statistical region merging (SRM) algorithm [121], which is based on region growing and merging. Lately, Ahn et al. [122] introduced Saliency based method to detect skin lesions by incorporating background detection coupled with inherent color characteristics in the dermoscopic images. In addition, they used hair removal as a pre-processing step. Bi et al. [123] presented a multi-scale superpixel with cellular automata (MSCA) to perform the segmentation of a skin lesion. As a pre-processing step, they adopted hair removal. Pennisi et al. [124] presented an approach called ASLM, which comprised four steps: 1) Artifact removal, 2) Skin detection, 3) Lesion segmentation, and 4) Merging the two images generated in steps 2 and 3 to produce the final segmentation map.

Active contours methods

They use curve evolution techniques through appropriate deformation to detect object contours [7]. Erkol et al. [125] developed a technique based on gradient vector flow (GVF) [126] to segment skin lesions in dermoscopy images. Zhou et al. [127] presented a new type of dynamic energy for skin lesion segmentation that combines a mean shift term [128] within the standard GVF model.

4.5.2 Traditional supervised methods

These methods need training data. They focus on extracting representative features, such as color, shape, or texture, by using several pre-processing techniques such as hair removal and illumination correction, and then train classifiers, such as support vector machines (SVMs) [129], wavelet network (WN) [130], to segment skin lesion.

Further details of all these techniques (unsupervised and traditional supervised methods) and other methods recently published are presented in [8, 131-139]. However,

the aforementioned methods do not capture high-level semantic information and only rely on low-level hand-crafted features (appearance information). Consequently, these techniques are still not robust and have difficulty for segmenting the challenging cases of low contrast, complex textures, and the presence of hair [5,9,10].

4.5.3 Deep learning-based methods

Deep learning-based methods, specifically CNNs, have recently become more popular for skin lesion segmentation.

Yu et al. [140] used a two-stage approach based on very deep residual networks (more than 50 layers), to acquire richer and more meaningful features, for the segmentation of skin lesions followed by classification. They evaluated their network on ISBI 2016 and were ranked second in segmentation and first in classification. Although the study obtained promising results, there were still some failure samples in situations such as low contrast, irregular shapes, and the presence of hair.

Bi et al. [141] proposed a multi-stage FCN to predict the results of segmentation in multiple stages. They used the parallel integration (PI) technique that enables the fusion of these segmentation results to better detect the lesion boundaries. By using post-processing techniques, such as morphological operation and connected thresholding, they achieved state-of-the-art results on the ISBI 2016 and PH2 datasets.

Yuan et al. [5] presented a FCN model with the introduction of a new loss function, based on the Jaccard distance to replace the conventionally used cross-entropy. They compared the segmentation performance using some key components of their model, such as input image size, data augmentation techniques, optimization method, and loss function. As post-processing operations, they used the dual-threshold method and morphological dilation. Their model outperformed the state-of-the-art methods on the ISBI 2016 and PH2 datasets. Despite these good results, there were some cases with suboptimal results.

Yuan et al. [142] introduced deep convolutional-deconvolutional neural networks (CDNN). In addition to the RGB channels, they have used additional channels from

multiple color spaces, of which three channels were from the HSV color space and other channel, which is the lightness channel (L) from the CIELAB color space. By using a bagging-type strategy to average the outputs of six models, they obtained the first place in the ISBI 2017 challenge.

Almasni et al. [9] used the full resolution features without the down-sampling path. Furthermore, they generated HSV images in addition to the RGB. Their approach outperformed other recent deep learning methods on ISBI 2017 and PH2 datasets. However, in terms of training times, this method was computationally expensive that took about 17.5 h even with using a powerful hardware (GPU of NVIDIA GeForce GTX 1080).

Mirikharaji et al. [143] proposed a deep auto-context fully convolutional neural network. They trained a sequence of FCNs in a consecutive manner, where the input of each network is the original image concatenated with the predicted segmentation map of the previous network.

Bi et al. [10] proposed three segmentation models, to segment non-melanoma, melanoma, and a last one for both non-melanoma and melanoma. They utilized a step-wise integration method to combine the segmentation results derived from each learning model. They used post-processing operations including morphological dilation and connected thresholding, to refine the binary segmentation result. They validated their method on ISBI 2017, ISBI 2016, and PH2 datasets, and achieved the superior performance compared to the existing works. However, training three models are computationally expensive, where each one took about 48 h using Nvidia Maxwell Titan X GPU.

Tang et al. [144] proposed a Separable-U-Net architecture, which uses the separable convolutional block in both the encoder and the decoder paths. To solve the overfitting problem, they introduced the stochastic weight averaging scheme as a solution.

Sarker et al. [145] introduced an encoder-decoder model. The encoder network consists of a dilated residual and a pyramid pooling network. They formulated a new loss function, which comprised Negative Log Likelihood (NLL) and End Point Error (EPE [146]) terms.

Shahin et al. [147] proposed an encoder-decoder network that uses pyramid pooling modules in the deep skip connections. They trained and validated their model using the ISIC 2018 dataset.

More recently, Hasan et al. [148] proposed an encoder-decoder called the dermoscopy skin network (DSNet). To minimize the number of network parameters, they used depth-wise separable convolution instead of standard convolution.

Öztürk and Özkaya [149] presented improved FCN (iFCN) structure. They used the elemental powers of different color spaces to eliminate the effect of disturbing factors, such as the presence of hair, illumination problems, and indistinct boundaries.

Xie et al. [150] introduced a novel CNN architecture, which generates high-resolution feature maps to preserve details, and adopted attention mechanism to highlight representative features while suppressing noise.

In [151], the authors proposed a segmentation model based on CNNs with using image representations from transform domain to improve the performance. In addition, CIELAB color space was concatenated with the input to provide more information for the model.

Zafar et al. [152] designed Res-Unet, which combines two networks, the U-Net and the ResNet. Moreover, they used image impainting for hair removal, to improve the segmentation performance.

Jiang et al. [153] presented an end-to-end framework based on U-Net structure and composed of the proposed CSARM modules (Channel and Spatial Attention Residual Module), multi-scale input layer, and side output layers. The new attention module CSARM combines channel attention mechanism, spatial attention mechanism, and residual learning to improve the model performance.

Several other CNN architectures have also been proposed in [154-170]. The following table (Table 4.2) presents implementation details of some of these methods, including datasets used, input image sizes, loss functions, and optimizers; and results in terms of JAC.

Ref.	Year	Input	Loss function	Optimizer	Dataset	JAC
[5]	2017	192×256	based on Jaccard	Adam	ISBI 2016	0.847
					PH2	-
[142]	2017	192×256	based on Jaccard	Adam	ISBI 2017	0.765
[167]	2021	120×160	Cross entropy (CE)	SGD	ISBI 2018	0.820
[169]	2021	192×256	Dice loss	Adam	ISBI 2017	0.7765
[166]	2021	256×256	Dice loss	Adam	ISBI 2018	0.8399
					ISBI 2017	0.7427
					ISBI 2016	0.8623
[170]	2019	-	CE	SGD	ISBI 2016	0.845
[159]	2021	224×224	CE	Adam	ISBI 2018	0.8330
[161]	2021	256×256	Dice loss +CE	Adam	ISBI 2017	0.7692
[148]	2020	192×256	CE+JD	Adadelta	ISBI 2017	0.775
					PH2	0.870
[150]	2020	512×512	Weighted CE	Adam	ISBI 2017	0.783
					ISBI 2016	0.858
					PH2	0.857

Table 4.2. Implementation details of some CNN architectures.

4.6 Conclusion

In this chapter, we briefly described dermoscopy, the principal layers of skin, and the common types of skin diseases. Then, we presented the common publicly available datasets of dermoscopy images, including ISBI 2018, ISBI 2017, ISBI 2016, and PH2, and the evaluation metrics for the segmentation task. Finally, we provided a comprehensive overview of methods used in skin lesion segmentation, including unsupervised methods, traditional supervised methods, and deep learning-based methods. In recent years, CNNs have attracted much attention and achieved promising results.

Chapter 5
Experimental Results

5.1 Introduction

In this chapter, we will describe our approaches, which are based on U-Net structure. The first approach is evaluated on ISBI 2016 dataset and compared to U-Net as a baseline. The second approach is evaluated on ISBI 2017 dataset and compared to the equivalent structure, a basic FCN (U-Net without skip connections) and U-Net as baselines. The third approach, which is the improvement of the first approach, is the main contribution of this thesis. It is evaluated on three datasets, including ISBI 2017, ISBI 2016 and PH2. This approach is compared to several state-of-the-art models, such as FCN, U-Net, SegNet, and U-Net++ as baselines and other latest existing methods in the literature.

5.2 Our approaches

5.2.1 A Modified U-Net for Skin Lesion Segmentation

In this approach, for skin lesion segmentation, we propose an encoder-decoder based on U-Net structure. The model employs dilated convolutional layers in both the encoder and the decoder to enlarge filter's field-of-view, in order to capture multi-scale information [20]. This network also uses pyramid pooling modules (PPMs) [17], which fuse features under multiple levels for more representativeness.

5.2.1.1 Network Architecture

The proposed model for automatic skin lesion segmentation is shown in Fig. 5.1b. The input is an RGB image, and the output is a probability map. Our model, like U-Net, is made up of two parts: a contracting path (encoder) for extracting abstract features and an expanding path (decoder) for recovering spatial resolution. The encoder comprises convolutional layers and max-pooling operations with a stride of 2. Convolutional layers extract the feature maps from the input image by convolving the latter with a set of kernels of size 3×3. We use dilated convolutional layers (with different dilation rates) to enlarge the receptive field of kernels. This strategy allows incorporating larger context without increasing the number of parameters associated with kernels. Max-pooling operations (downsampling) are used to reduce the size of the extracted feature maps by only retaining a pixel with the largest value among the neighboring four pixels. This aims to be efficient

in memory albeit with a loss of the spatial resolution of the feature maps. The last level of the encoder contains two standard convolutional layers followed by a pyramid pooling module (PPM) and two other standard convolutional layers. The PPM collects information at multiple levels for more representation of features [17]. On the other hand, the decoder is built by alternating of 3×3 deconvolution (transposed convolution) that halves the number of features maps, a skip connection, and two 3×3 dilated convolutions (except the second level, which has three 3×3 dilated convolutions). Deconvolution is used to increase its input size (upsampling). Similar to U-Net, to recover the spatial information lost by pooling operations, we use skip connections. These concatenate the low-level feature maps of the contracting path with the corresponding feature maps of the expanding path, except for the third encoding stage, where the feature maps are first input into the PPM block, and then the output features are concatenated with the corresponding feature maps of the decoder. These concatenated feature maps are then convolved with a set of kernels of size 3×3 to produce dense feature maps. To generate the output segmentation map, the decoder is then followed by a 1×1 convolution and the sigmoid layer as a pixel-wise classification. A Batch Normalization (BN) layer [44] and a Rectified Linear Unit ($ReLU$) activation function, follow each 3×3 convolution to alleviate the vanishing gradient problem, and accelerate the training process.

A. Dilated convolution

In semantic segmentation, downsampling causes loss of spatial information of feature maps. To overcome this limitation, Yu and Kotlun [16] adopted the dilated convolutions to aggregate multi-scale contextual information without reducing spatial resolution. The main idea of the dilation is to upsample convolutional kernels by inserting "holes" (zeros) between kernel values, thus enables to maximize features extraction ability with an enlarged receptive field, and without extra parameters. Fig. 5.2 illustrates the different dilation rates adopted $(D = 1, 2, 4)$ to increase the receptive field of our model. The receptive field of kernel k with a size $N \times N$ can be defined as

$$R_k = N + (N - 1)(D - 1) \tag{5.1}$$

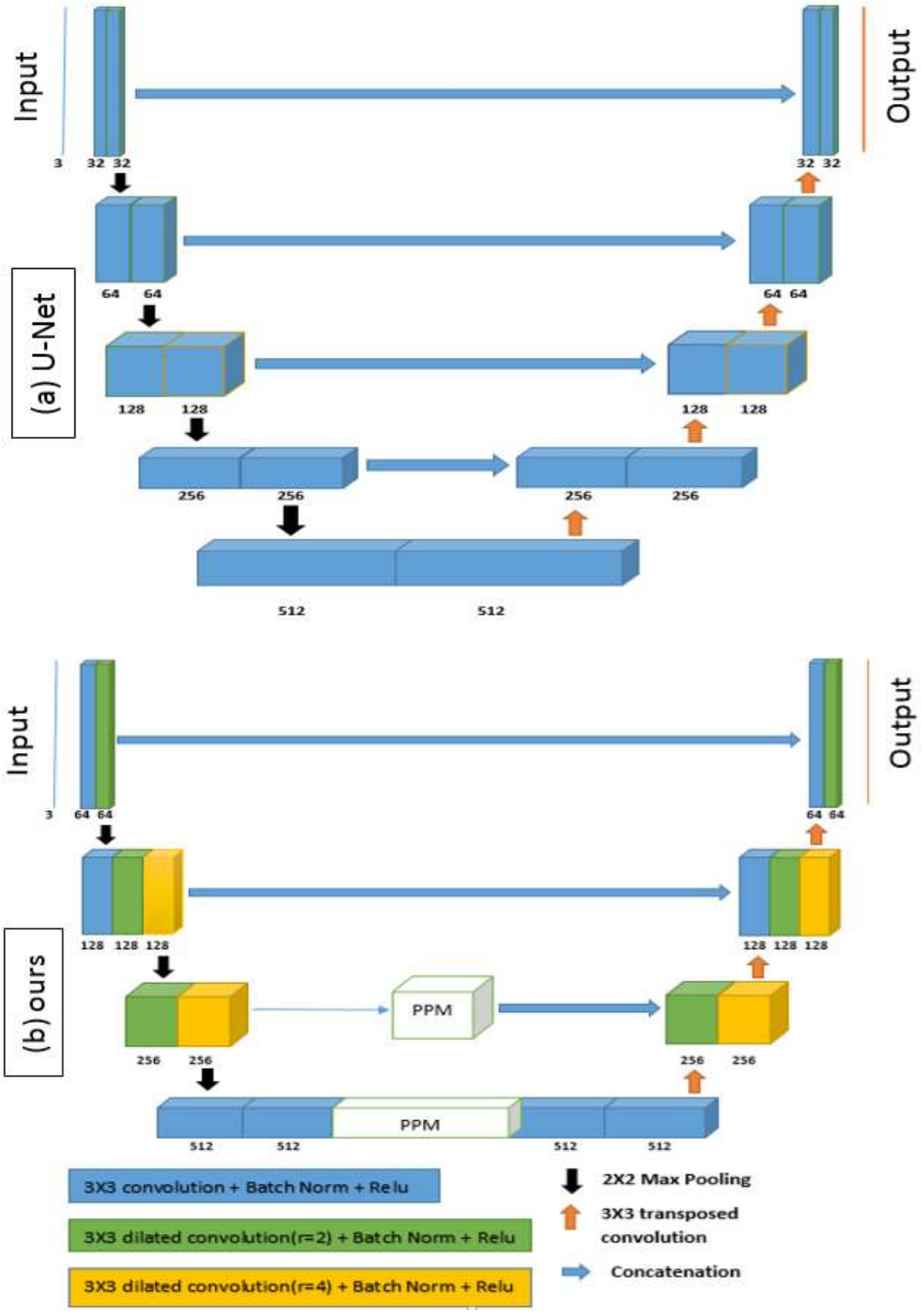

Figure 5.1. Overview of the U-Net (a) and the proposed architecture (b). The number of feature maps is denoted at the bottom of each convolutional layer. Layers with the same color correspond to the same dilation rate.

In this work, $N = 3$ (we fixed the size of kernels). D is the dilation rate specifying the number of zeros between kernel values. Note that in normal convolution, $D = 1$.

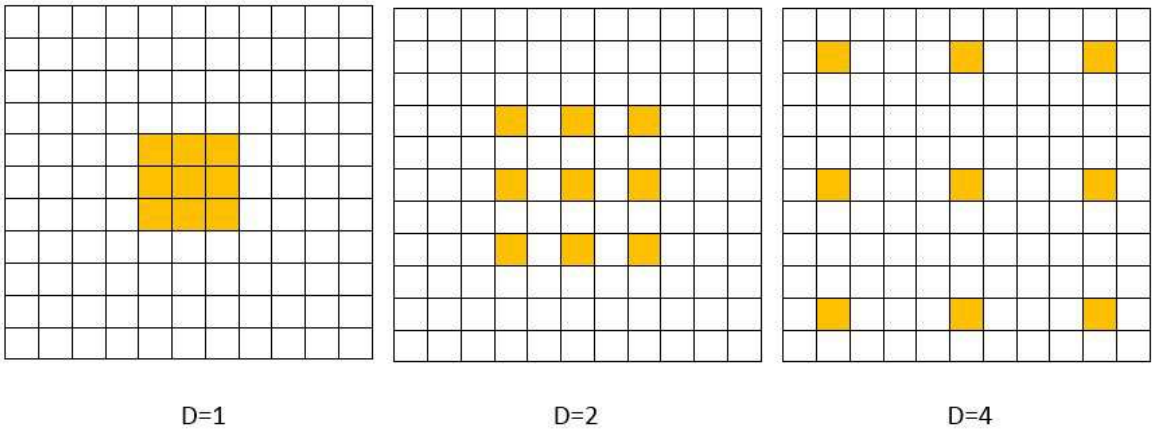

Figure 5.2. Dilated convolution with different dilation rates D. Note that in cases of $D = 1, D = 2$, and $D = 4$, the receptive field of a kernel of size of 3×3 will be 3, 5 and 9, respectively. Dilated convolution expands the receptive field, without losing spatial resolution, and without extra parameters

B. Pyramid Pooling Module

Inspired by PSPNet [17], in order to obtain more contextual information, we perform pooling operations at multiple grid scales. We integrate the pyramid pooling module (PPM) at the skip connection in the third encoding stage, and between the encoder and the decoder as shown in Fig. 5.1b. As illustrated in Fig. 5.3, the PPM has four levels in parallel. Each level employs a pooling layer with a different ratio. As a result, feature maps with different sizes are obtained. Note that the pooling sizes and the number of pyramid levels can be modified, depending on the size of the feature maps fed into the PPM. We use a 1×1 convolutional layer after each pyramid level to reduce the number of feature maps to $1/4$ of the input features (there are 4 levels). Then, we upsample the low-dimension feature maps to the original sizes via transposed convolution. The latter uses a set of trainable kernels to gather more information. Finally, upsampled feature maps of all levels are concatenated with the original input features. For the PPM at the skip connection, the pooling sizes in the four levels are 1×1 max-pooling, 2×2 max-pooling, 4×4 max-pooling, and 8×8 max pooling, respectively. For the other PPM, in the four levels, we use

1×1 max-pooling, 2×2 max-pooling, 3×3 max-pooling, and 6×6 max-pooling, respectively.

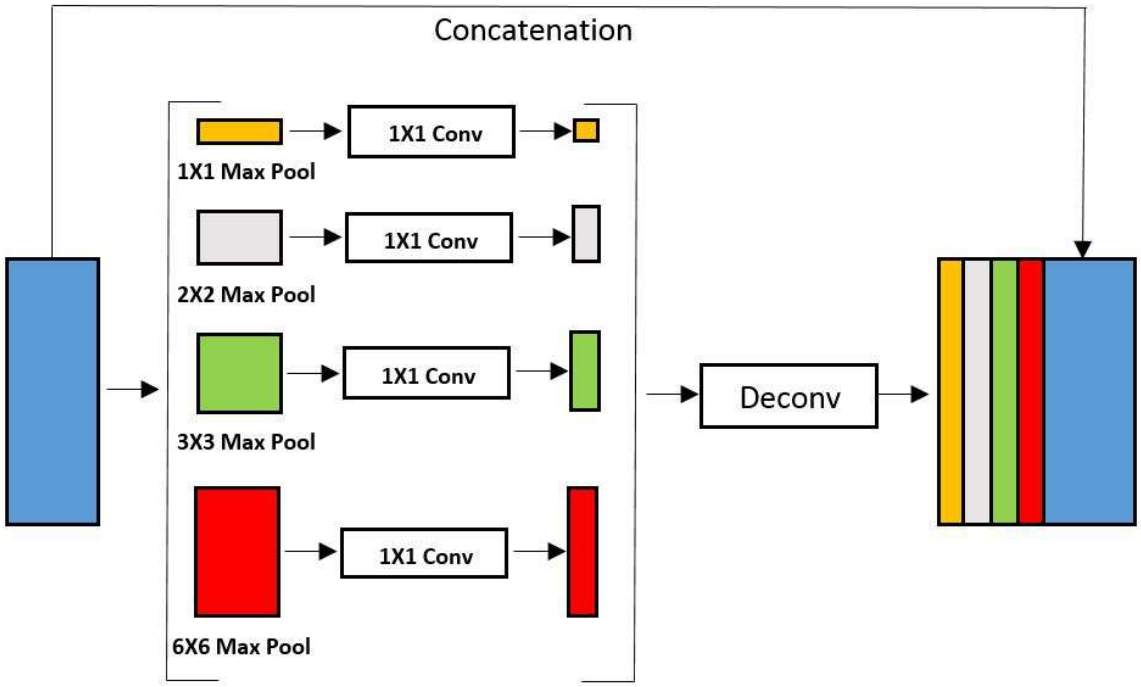

Figure 5.3. Pyramid Pooling Module (PPM) with four levels in parallel and deconvolutional layer (3×3 transposed convolution) for upsampling

C. Loss Function

The loss function measures the error between the predicted segmentation maps and their ground truth labels. To alleviate the problem of class imbalance (small foreground lesions in a large background skin), we use a loss function based on Jaccard distance [5], which is described as:

$$L_{d_J} = 1 - \frac{\sum_{i,j}(t_{ij}p_{ij})}{\sum_{i,j}{t_{ij}}^2 + \sum_{i,j}{p_{ij}}^2 - \sum_{i,j}(t_{ij}p_{ij})} \tag{5.2}$$

Where t_{ij} and p_{ij} denote the ground truth and the predicted class of pixel (i, j), respectively.

5.2.1.2 Experiments

A. Database

We experimented on ISBI 2016 [99], Skin Lesion Analysis Towards Melanoma Detection Challenge dataset. The training set consists of 900 images. We used 80% for training (720 images) and 20% for validation (180 images). For evaluation, another set of 379 images and their ground truths were provided. All images were resized to 192×256 pixels to reduce the computational cost, and normalized to values between 0 and 1. To reduce model overfitting, we augmented the training images using various geometric transformations, such as rotation, horizontal flipping, vertical flipping, and zooming.

B. Baseline

The performance of our model was compared with that of the U-Net as a baseline. For U-Net, We employed in the first layer 32 kernels instead of 64 in the original version [14]. To evaluate our model, we used Jaccard index (JC) as the main metric of ISBI 2016. We also employed Dice (DIC), sensitivity (SEN), specificity (SPE), and accuracy (ACC) metrics to calculate the segmentation results.

C. Implementation

We implemented our model and U-Net using Keras, with Tensorflow backend and trained on Google colab[1] with Tesla K80 GPU. We used the Adam optimizer [42] as the optimization algorithm with an initial learning rate of 0.0001 and then decreased it by half each time encountering 5 epochs without improvement on the validation set. We also employed "he-normal" scheme [52] to randomly initialize the model weights and early stopping mechanism on the validation set. The batch size was set to 16 and the epoch to 150.

D. Results

[1] https:colab.research.google.com

We monitored the progress of Jaccard index with the number of epochs. The result on the training data is shown in Fig. 5.4a. The performance on the validation data is shown in Fig. 5.4b. It can be noted that for both cases, our model shows better performance when compared to the U-Net.

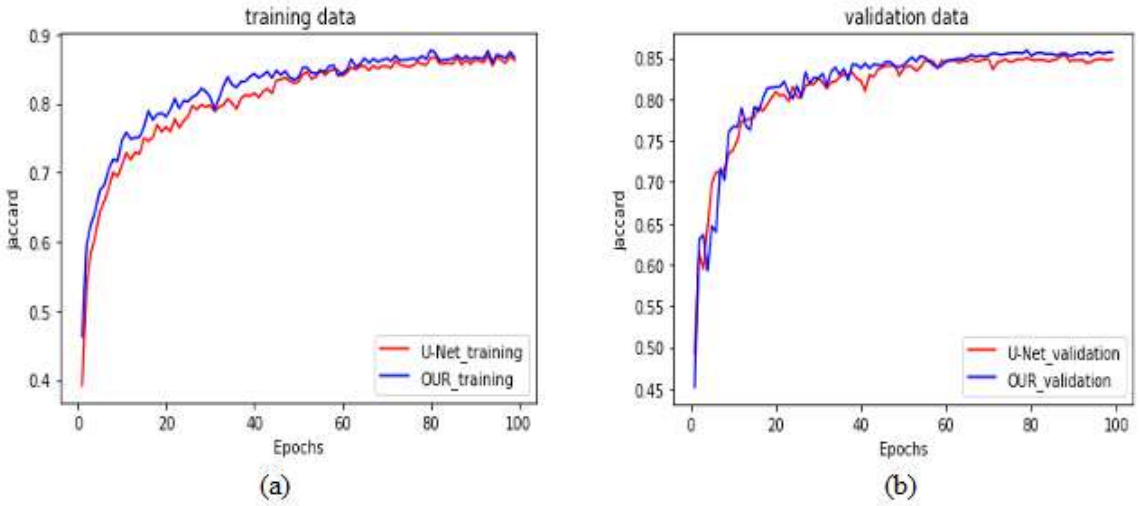

Figure 5.4. Performance on training data (a) and validation data (b)

The quantitative results on the test data is shown in Table 5.1. We compared the proposed model to the U-Net (the baseline) and another published method on the official ISBI 2016 test set (ranked third), with respect to the evaluation metrics of ISBI 2016 including Jaccard index , Dice , sensitivity , specificity , and accuracy . From the Table 5.1, it can be observed that our model provides better results than the other methods in terms of three evaluation metrics $(JAC = 82.7, DIC = 89.6, SEN = 92.0)$.

Data	Methods	Year	JAC	DIC	SEN	SPE	ACC
ISBI 2016	Rahman et al [171]	2016	82.2	89.5	88.0	**96.9**	**95.5**
	U-Net (Baseline)	2019	82.3	89.4	91.7	95.5	93.7
	Our model	2019	**82.7**	**89.6**	**92.0**	95.2	93.9

Table 5.1. Quantitative results on test data.

Qualitative results of some challenging samples from the test set of ISBI 2016 (without any post-processing) are shown in Fig. 5.5. From the left to right, each column represents the input image, the ground truth, the segmentation map generated by U-Net, and the segmentation map generated by our model, respectively. The first and second rows show the success cases of our model over U-Net (the results of our model have smaller sizes of false negative gaps). The third and fourth rows show the failure cases of the proposed network over U-Net (the results of our model have bigger sizes of false positive gaps).

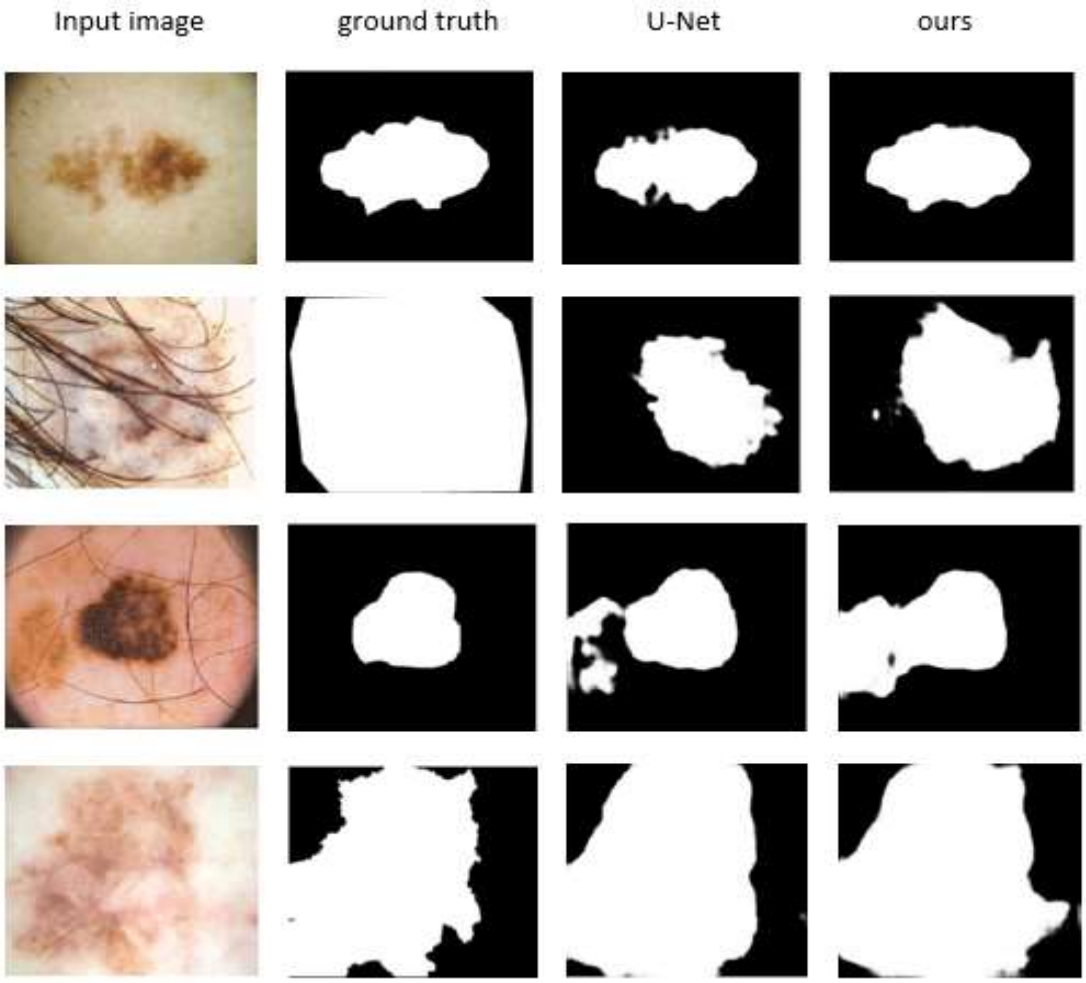

Figure 5.5. Qualitative results of some challenging samples.

5.2.2 FCN-MASPP: Fully Convolutional Network with Modified Atrous Spatial Pyramid Pooling modules for skin lesion segmentation.

Motivated by the success of U-Net structure [14], the ASPP module [18, 19, 20], and the PPM [17], we propose a novel fully convolutional network called FCN-MASPP. The main idea is to integrate the MASPP module into each level of both the encoding and decoding paths to learn features that are more representative.

5.2.2.1 Overview

Inspired by the original U-Net structure [14], the proposed model is a symmetric architecture and has five levels, as illustrated in Fig. 5.6. It consists of encoding and decoding paths followed by a sigmoid activation function as a pixel-wise classification. The encoding path comprises 5 MASPPs, 10 convolutional layers (3×3 convolution), and 4 pooling layers (2×2 max-pooling). On the other hand, the decoding path is composed of 4 upsampling layers (2×2 transposed convolution), 4 MASPPs, and 8 convolutional layers. Batch normalization (BN) and rectified linear unit ($ReLU$) activation follow each 3×3 convolution. As shown in Fig. 5.6, our network shares a similar architecture to the original U-Net, but with some differences. First, instead of using only 3×3 convolutional layers like U-Net, we add MASPP module and integrate it as input into each level of both the encoder and decoder. Second, unlike U-Net, the proposed FCN-MASPP has no skip connections between the encoder and the corresponding decoder. We note that the skip connections between the encoder and the decoder were tested but without further improvement.

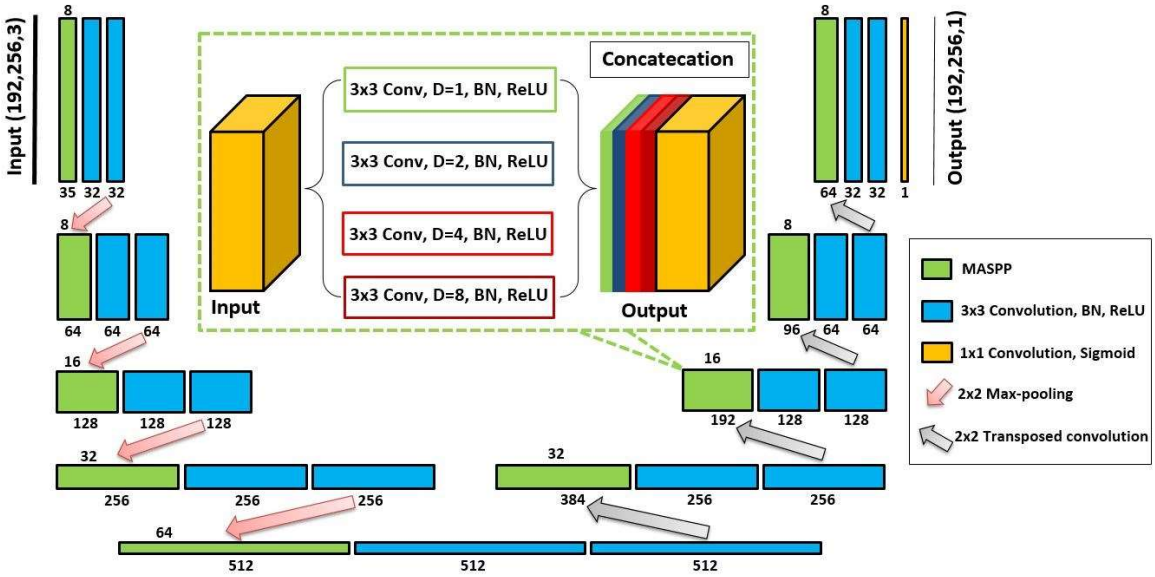

Figure 5.6. The structure of the proposed FCN-MASPP. The input is an RGB image and the output is a probability map. The number of kernels assigned to each level of the MASPP is shown above each MASPP box. The number of output feature maps is shown below each box. Note that like U-Net, each 2x2 transposed convolution halves the number of feature maps.

A. MASPP

Inspired by DeepLab [18], to capture multi-scale contextual information at each level of the network, we propose a modified atrous spatial pyramid pooling (MASPP) module, which is integrated at each level of both the encoding and decoding paths, as shown in Fig. 5.6. The original atrous spatial pyramid pooling (ASPP) proposed by DeepLab is shown in Fig. 3.24. It consists of multiple levels of dilated (atrous) convolution in parallel with different dilation rates. The proposed MASPP has four levels of atrous convolution in parallel, where the dilation rate in the four levels is 1, 2, 4, and 8, respectively. The number of kernels assigned to each level of the MASPP is 8, 8, 16, 32, and 64 for the five encoder levels, respectively, while it is 32, 16, 8, and 8 for the four levels of the decoder, respectively. Note that the dilation rate, the number of kernels applied at each level of the MASPP, and the number of MASPP levels can be modified. We choose this configuration to reduce the parameter space. As a result, multi-scale contextual feature maps are obtained. Inspired by PSPNet [17], these features are concatenated with the original input features and fed into a next layer, as illustrated in Fig. 5.6.

5.2.2.2 Results and Discussions

A. Dataset

We used the well-established public benchmark ISBI 2017 dataset [100] in the evaluation of the proposed model.

For the preprocessing, we first resized all the images into 192×256 pixels and normalized each RGB pixel value to [0, 1]. At the testing stage, the output segmentation maps were resized to the size of the original images to perform a quantitative comparison according to the evaluation metrics.

B. Evaluation metrics

We adopted the evaluation indicators suggested by ISBI 2017 challenge, including Jaccard index (JAC) as the main metric for ranking, dice coefficient (DIC), sensitivity (SEN), specificity (SPE), and accuracy (ACC).

C. Implementation

The experiments were conducted in a desktop computer with the following configuration: Intel® Core (TM) i7-8700K CPU @ 3.70 GHz, with 16 GB RAM, and GPU NVIDIA GeForce GTX 1070. The proposed model FCN-MASPP and baselines were implemented with python 3.6.10, Using Keras, and Tensorflow backend.

To address the class imbalance issue, we trained all models using Tversky loss (TL) [172, 173] defined as:

$$TL = 1 - \frac{\sum_{i,j}(p_{1ij}t_{1ij})+\epsilon}{\sum_{i,j}(p_{1ij}t_{1ij})+\alpha\sum_{i,j}(p_{0ij}t_{1ij})+\beta\sum_{i,j}(p_{1ij}t_{0ij})+\epsilon} \quad (5.3)$$

Where t_{1ij}, p_{1ij} are the ground truth and the prediction probability of pixel (i,j) being in a lesion region; and t_{0ij}, p_{0ij} represent the ground truth and the prediction probability of pixel (i,j) being in a healthy skin region. α and β (where $\alpha + \beta = 1$) are adjustable

parameters that control the magnitude of penalties for FNs and FPs, respectively. ϵ is added to avoid zero in denominator. In this work, we set ($\alpha = 0.7, \beta = 0.3, \epsilon = 10^{-7}$).

We used Adam [42] as the optimization algorithm with an initial learning rate of 0.0001. Then, during training, the learning rate was decreased by half that of the previous one (until: $1.25*10^{-5}$) if no improvement has been noticed for 10 epochs on validation data. The network parameters (weights) were initialized using a "he_normal" scheme [52]. We adopted the not additive data augmentation technique by using geometric transformations to augment the training set online. These geometric transformations included rotation, horizontal flipping, vertical flipping, and zooming. The batch size used was 8 and the epochs were set to 200. Finally, to accelerate the training stage, early stopping mechanism on the validation set was performed. This strategy interrupts training when no progress is made after 40 epochs.

D. Analysis of results

The quantitative results obtained are shown in Table 5.2. We used the U-Net, the basic FCN (U-Net without skip connections) as baselines. As shown, the proposed model with a powerful module MASPP has better results in all metrics, except in SEN, when compared to baselines (basic FCN and U-Net).

Model	JAC	DIC	SEN	SPE	ACC	parameters
Basic FCN	0.7622	0.8484	0.8763	0.9591	0.9322	6,982,625
U-Net	0.7615	0.8472	**0.8820**	0.9563	0.9315	7,765,985
FCN-MASPP (ours)	**0.7769**	**0.8587**	0.8795	**0.9608**	**0.9387**	10,137,665

Table 5.2. Results on ISBI 2017 test dataset.

As shown in Table 5.2, due to the use of MASPP modules, the proposed model has a higher size of parameter space, but this amount of parameters is acceptable.

Qualitative results of some challenging examples are shown in Fig. 5.7 and Fig. 5.8. As can be observed, FCN-MASPP exhibits superior performance when compared to a basic FCN and U-Net. Figs. 5.7a-c and 5.7d-h demonstrate the capability of the proposed model to segment with smaller sizes of FNs and FPs, respectively.

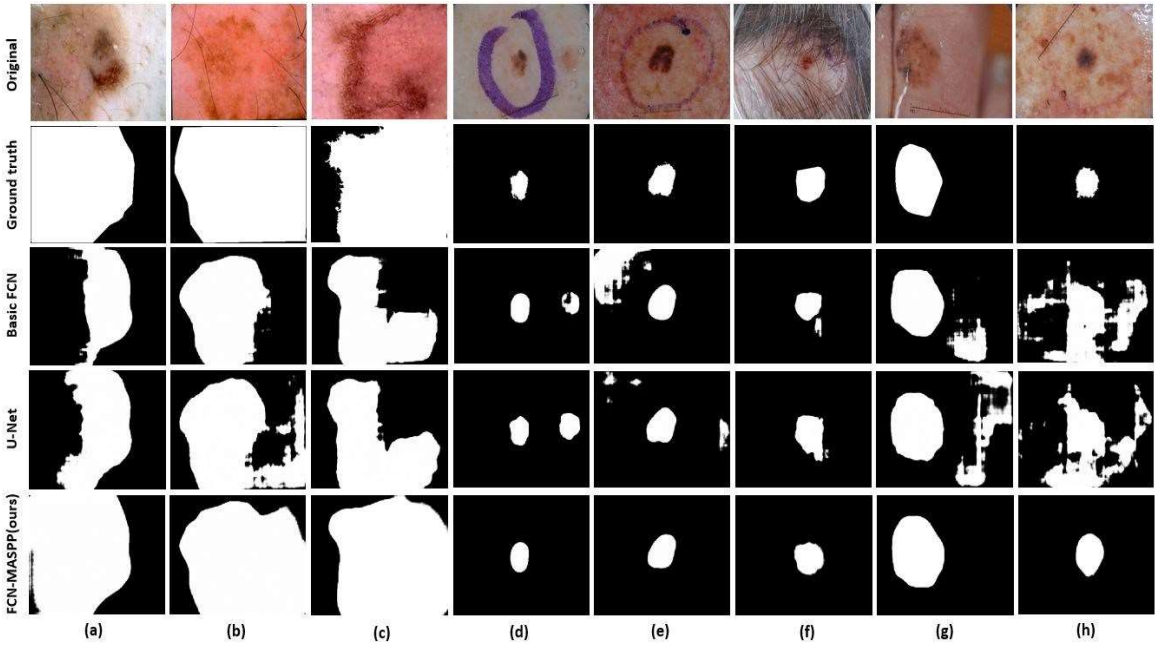

Figure 5.7. Segmentation results of some challenging examples of ISBI 2017 test dataset.

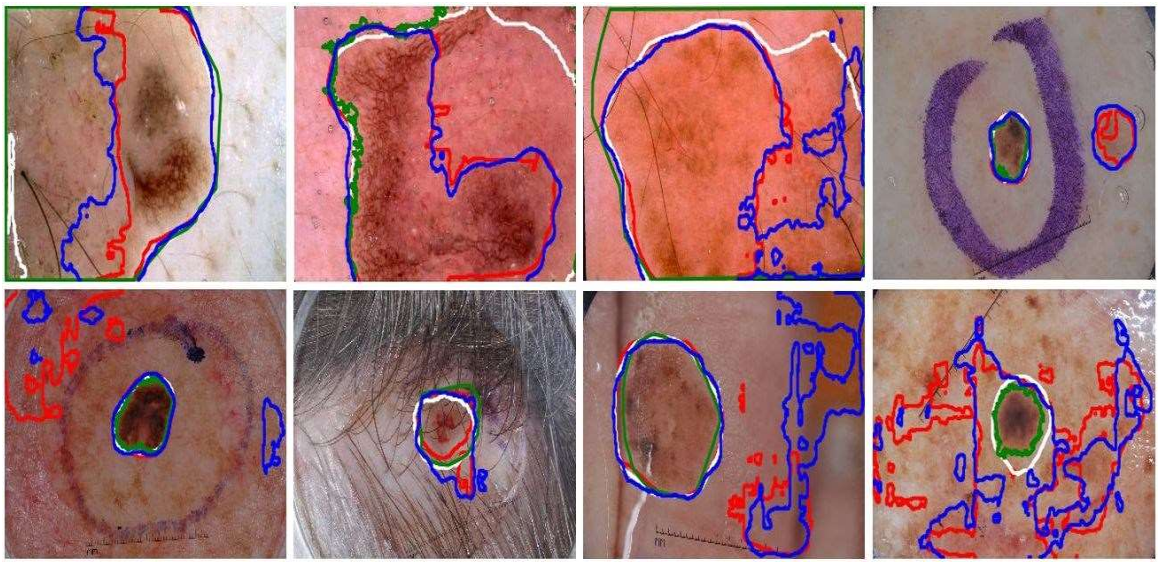

Figure 5.8. Comparison of segmentation results. The green, red, blue, and white contours represent the ground truth, the result of a basic FCN, the result of U-Net, and the result of FCN-MASPP, respectively.

5.2.2.3 Conclusion

In this approach, we proposed a novel FCN called FCN-MASPP, for skin lesion segmentation. The main component of FCN-MASPP is the novel MASPP, which is

integrated into each level of both the encoder and decoder. The integration of MASPP as such further enhances the features extraction ability by learning multi-scale context information at each level of the network. The proposed FCN-MASPP outperformed the equivalent structure, a basic FCN and U-Net on the challenging ISBI 2017 skin lesion dataset.

5.2.3 An Improved and Robust Encoder–Decoder for Skin Lesion Segmentation

In this approach, we propose major modifications to the state-of-the-art U-Net structure to further improve its capability in skin lesion segmentation while keeping its efficiency in terms of computational time for both the training and testing stages. These modifications are presented in both the encoding and the decoding paths. Instead of using only standard convolutional layers like U-Net, the proposed encoding path consists of 10 standard convolutional layers, which are inspired by the Visual Geometry Group (VGG16) network [22], followed by a pyramid pooling module and a dilated convolutional block. This combination enables to learn better representative feature maps and preserve more spatial resolution. Furthermore, dilated residual blocks are introduced in the decoding path to further refine the segmentation maps. The experimental results on three datasets, including the IEEE International Symposium on Biomedical Imaging (ISBI) 2017, ISBI 2016, and PH2, showed that our proposed method has better performance than the basic U-Net [14], FCN [23], SegNet [24], and U-Net ++ [25], and achieves the performance of state-of-the-art segmentation techniques with minimum pre- and post-processing operations.

5.2.3.1 Overview of the Proposed Method

The proposed segmentation model is shown in Fig. 5.9. This architecture is composed of two parts: an encoding path and a corresponding decoding path, followed by a sigmoid layer as a pixel-wise classification. The encoding path extracts abstract features from the input RGB image, while the decoding path gradually recovers features spatial resolution. The proposed network is different from U-Net in both the encoding and decoding paths. U-Net is a symmetric architecture in which both the encoding and the decoding paths consist of a sequence of two standard convolutional layers, while the encoding path in our model contains three different components: the first 10 convolutional layers of VGG16

[22], a pyramid pooling module (PPM) [17], and a dilated convolutional block (DCB). On the other hand, we replace the standard convolutional layers in the decoding path with the dilated residual blocks (DRBs) to further refine the segmentation map. In the following, we will discuss about the proposed encoding and decoding paths.

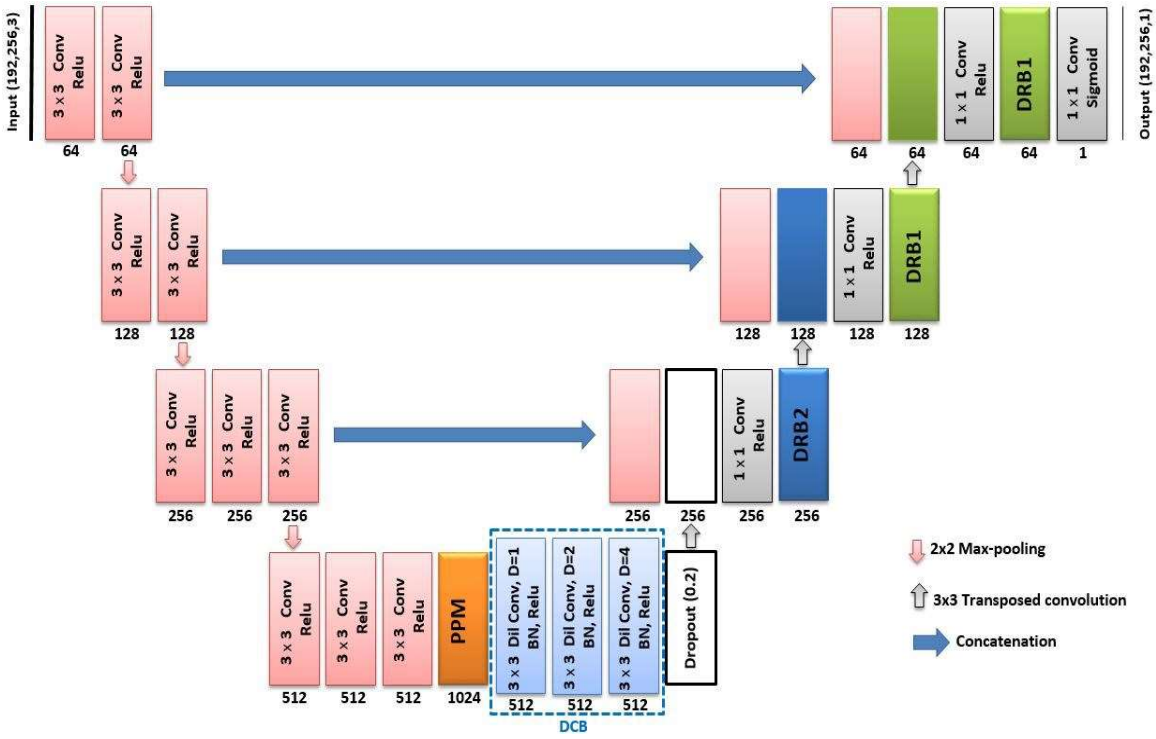

Figure 5.9. Overview of the proposed architecture. The encoding path starts with the first 10 convolutional layers of the VGG16 network (left side). The input is an RGB image ($192 \times 256 \times 3$), while the output is a probability map ($192 \times 256 \times 1$). The number of feature maps is denoted at the bottom of each convolutional layer

A. The Encoding Path

The first component of the encoding path consists of 10 standard convolutional layers, which are inspired by the VGG16 network. A Rectified linear unit ($ReLU$) activation function follows each 3×3 convolution, and 2×2 max-pooling operation is sometimes performed between convolutional layers. In our method, we just choose the first 10 layers of VGG16 network to reduce the parameter space and preserve acceptable spatial

information of feature maps. To learn more representative features, these 10 convolutional layers are followed by a pyramid pooling module (PPM) and a dilated convolutional block (DCB). Finally, we add a dropout layer with a probability $p = 0.2$ to reduce model overfitting. During the training process, this layer randomly drops some nodes and their connections to alleviate overfitting to the training images.

PPM

To capture information at multiple levels and detect lesions of different sizes, we perform multi-scale pooling operations. As illustrated in Fig. 5.3, the PPM consists of four levels in parallel, where the pooling size in the four levels is 1×1 max-pooling, 2×2 max-pooling, 3×3 max-pooling, and 6×6 max-pooling, respectively. The output feature maps of the PPM are fed into a DCB to further capture multi-scale contextual information.

Dilated Convolutional Block (DCB)

The used DCB illustrated in Fig. 5.9, groups three dilated convolutional layers with batch normalization (BN) and $ReLU$ activation in each one. BN [44] normalizes each training mini-batch to reduce the internal covariate shift. To alleviate the "gridding issue" [71, 174-176], the dilation rate D is exponentially increased with values of 1, 2, and 4 for the three convolutional layers, respectively.

B. The Decoding Path

The decoding path is built by alternating of 3×3 transposed convolution (that halves the number of feature maps), a skip connection, 1×1 convolution, and dilated residual block (DRB). Transposed convolution with learnable parameters is used for upsampling. Similar to U-Net, to restore the spatial information lost by pooling layers, skip connections are performed by concatenating the low-level feature maps of the encoder with the corresponding feature maps of the decoder. Then, we use 1×1 convolution as a bottleneck to reduce the number of feature maps. Finally, the dilated residual block (DRB) is proposed to extract more context information and further refine the segmentation map. The DRB stacks dilated convolutional layers with residual connections. These residual connections alleviate the vanishing gradient problem and facilitate the training process,

since a DRB learns a function with reference to the layer inputs, instead of learning unreferenced functions [62].

The DRBs are shown in Fig. 5.10, where each one (DRB1 and DRB2) can be expressed as follows:

For DRB1:

$$x_{1o} = H_1(x_{in}) + x_{in} \tag{5.4}$$

For DRB2:

$$x_{2o} = H_2(ReLU(x_{1o})) + x_{in} \tag{5.5}$$

Where x_{in}, x_{1o}, x_{2o} are the input, the output of the DRB1, and the output of the DRB2, respectively. The residual function H_1 consists of two 3×3 dilated convolutional layers. The first layer (with D = 1) is followed by (BN) and a $ReLU$ activation function, whereas the second layer (with D = 2) is only followed by BN. H_2 is a 3×3 dilated convolutional layer (with D = 4) followed by BN. The DRB2 comprises two skip connections (via addition) to further ease the information propagation during the back pass of back-propagation.

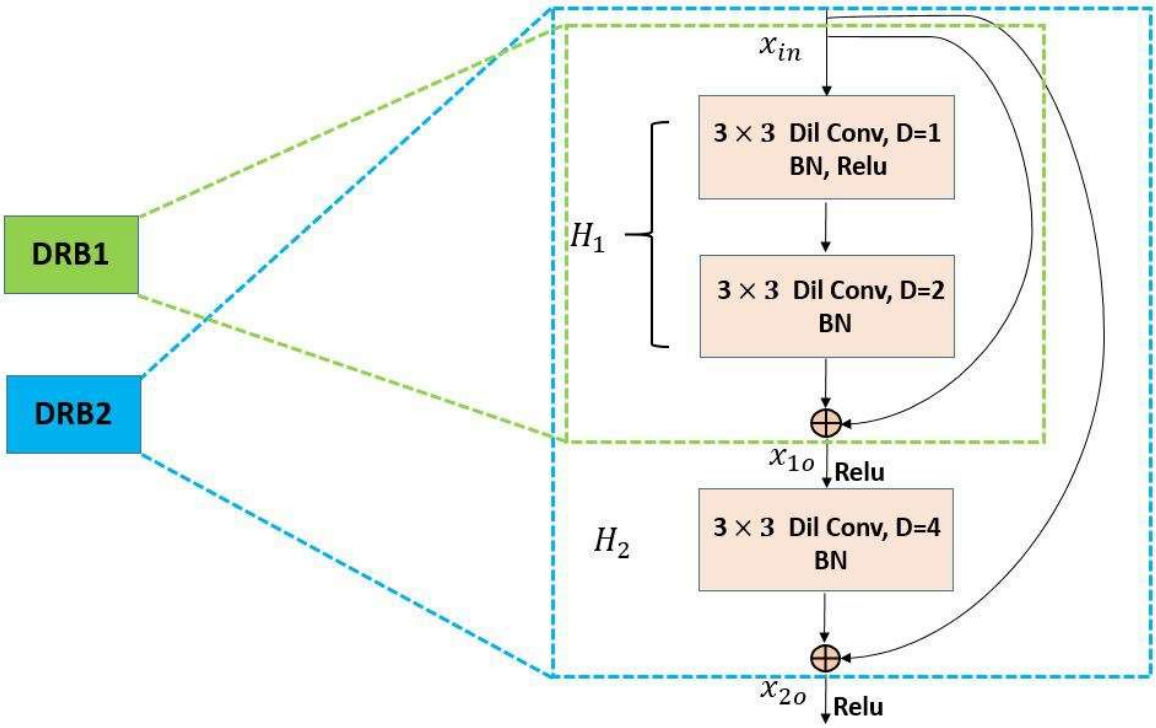

Figure 5.10. Dilated Residual Blocks (DRBs). DRB1 includes two dilated convolutional layers, and one skip connection (via addition), while DRB2 consists of three dilated convolutional layers and two skip connections (via addition)

5.2.3.2 Materials and Implementation Details

The proposed model was implemented with python 3.6.10 using Keras, and Tensorflow backend. The experiments were conducted in a desktop computer with the following configuration: Intel® Core(TM) i7-8700 K CPU @ 3.70 GHz, with 16 GB RAM, and GPU NVIDIA GeForce GTX 1070.

A. Datasets

We used three public benchmark datasets to evaluate our proposed method. These are ISBI 2016, ISBI 2017, and PH2.

B. Evaluation Metrics

To evaluate the proposed model, we used the following standard evaluation metrics of ISBI 2016 and ISBI 2017 challenges, including Jaccard index (JAC), dice coefficient

(DIC) , sensitivity (SEN) , specificity (SPE), and accuracy (ACC). In addition, we calculated the Matthew correlation coefficient (MCC), which is defined as:

$$MCC = \frac{TP.TN - FP.FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (5.6)$$

C. Training and Testing

The objective of the training is to find model weights that best predict input images by minimizing a loss function. We used a loss function based on Jaccard distance [5]. As an optimization algorithm, we used Adam [42] to adaptively adjust the learning rate based on the first and the second moments of the gradient. To speed up the training process, the initial learning rate was set to 0.0001. For better optimization, the learning rate was then decreased by half (until: $1.25*10^{-5}$), each time encountering 10 epochs without improvement on the validation set. The only pre-processing applied is that all input images were resized into 192×256 and normalized. The aim of resizing is to reduce the computational cost and deal with memory constraints. We chose this resolution to preserve the aspect ratio, since most of the images in the training datasets have an aspect ratio (height-to-width) of $3{:}4$. Resizing was carried out using bilinear and nearest interpolations for images (training, validation, and test datasets) and their ground truths, respectively. On the other hand, the goal of normalization is to bring the input pixel values to the same range (values between 0 and 1) by dividing them by 255. The standard convolutional layers of the network were initialized using the pre-trained weights on ImageNet dataset [55], while we employed a "he_normal" scheme [52] to randomly initialize the rest of the model parameters (weights)."he_normal" scheme takes samples from a zero-mean truncated normal distribution with a standard deviation (std) of $\sqrt{2/m}$, where m is the number of input units in the weight tensor. We adopted online data augmentation to augment the training images using various geometric transformations including rotation, horizontal flipping, vertical flipping, and zooming. This technique is not additive and just replaces the original training images with the randomly transformed. The batch size was set to 8 to keep balance between memory consumption and computation cost, while the epochs were set to 200. Early stopping mechanism on the validation set was performed to interrupt training

when no further improvement was noticed after 40 epochs. The model inputs are the batches of training and validation datasets (with corresponding ground truths for each one). For experiments using the 2017 dataset, we adopted the specified training and testing dataset (2000 images for training, 150 images for validation, 600 images for testing), while the pre-trained model on ISBI 2017 was used for the evaluation on the ISBI 2016 test and PH2 datasets. As the batches of the training are selected, a set of geometric transformations are applied at runtime. Training continues until the early stopping mechanism is satisfied. At each iteration, the model parameters are updated using Adam optimizer via back-propagation. After each epoch, the loss function is measured on the validation dataset, and the model weights are saved if an improvement is noticed. When training is complete, the final saved weights are used at the testing stage. Due to the use of an early stopping mechanism, the training took about 2.64 h over 84 epochs for the ISBI 2017 dataset.

At the testing stages, a threshold of 0.5 was applied to the output segmentation map. Then, a morphological dilation and erosion were used for ISBI 2017 test dataset and PH2 dataset, respectively. These post-processing operations improved the results on ISBI 2017 test dataset (in terms of JAC, DIC, SEN, and ACC) and PH2 dataset (in terms of JAC, DIC, SPE, and ACC) as shown in Tables 5.4 and 5.6. For ISBI 2016 test dataset, we did not apply any post-processing since no improvement was noticed. This may be due to that ISBI 2016 dataset is less challenging than ISBI 2017 and PH2 datasets. Finally, resizing to the size of the original image was carried out by using the nearest interpolation.

5.2.3.3 Experimental Results

In our experiments, to investigate how the VGG16 convolutional layers affect the performance of the model, we implemented 3 models, defined as:

Model-7: the model just has the first 7 standard convolutional layers of VGG16 network in its encoding path.

Model-10 (proposed): the model has the first 10 standard convolutional layers of VGG16 network in its encoding path.

Model-13: the model uses all VGG16 convolutional layers (13 layers) in the encoding path.

The results summarized in Table 5.3 show that the Model-13, which has the highest number of parameters (37.5 M), achieved the best performance on ISBI 2017 with JAC of 0.7812. It was ranked third when tested on the ISBI 2016 and PH2 dataset, with JAC of 0.8597 and 0.8514, respectively. The Model-7, which has the lowest number of parameters (10.7 M), had a higher performance on ISBI 2016 with JAC of 0.8714. When tested on the PH2 and ISBI 2017 test datasets, it was ranked second with JAC of 0.8514 and third with JAC of 0.7471, respectively. The Model-10 obtained the superior performance on PH2 with JAC of 0.8616. Furthermore, it achieved the second best performance on other datasets, with overall JAC indices of 0.7789 and 0.8639 on ISBI 2017 (after Model-13) and ISBI 2016 (after Model-7) test datasets, respectively. These results indicate the robustness and generalization capability of the proposed model (Model-10).

Method		*JAC*	*DIC*	*SEN*	*SPE*	*ACC*	*MCC*	Parameters
ISBI 2017	Model-7	0.7471	0.8335	0.8345	**0.9721**	0.9278	0.8024	10,712,577
	Model-10	0.7789	0.8601	0.8745	0.9621	0.9375	0.8262	21,792,897
	Model-13	**0.7812**	**0.8636**	**0.8808**	0.9601	**0.9379**	**0.8290**	37,528,449
ISBI 2016	Model-7	**0.8714**	**0.9278**	0.9273	**0.9748**	**0.9647**	**0.8958**	
	Model-10	0.8639	0.9233	0.9311	0.9676	0.9621	0.8902	
	Model-13	0.8597	0.9208	**0.9354**	0.9623	0.9594	0.8863	
PH2	Model-7	0.8533	0.9167	0.9280	**0.9630**	0.9417	0.8640	
	Model-10	**0.8616**	**0.9222**	0.9459	0.9551	**0.9470**	**0.8738**	
	Model-13	0.8514	0.9158	**0.9507**	0.9441	0.9438	0.8643	

Bold values indicate the best results

Table 5.3. Influence of the VGG16 layers on the performance of the model.

A. Ablation study

In order to assess the contributions of our network and justify the design choice, the following ablation studies have been conducted on the most challenging dataset ISBI 2017:

1) We used as the backbone architecture the encoding path (the 10 conventional convolutional layers + PPM + DCB), with the corresponding decoding path that

replaces the DRBs with conventional convolutional blocks (CCBs), without residual connections. The model is denoted as ‘’Backbone-CCBs’’ in Table 5.4.

2) We removed the residual connections in DRBs. As such, the decoding path consists of dilated convolutional blocks (DCBs), and the model is referred to as‘’Backbone-DCBs’’in Table 5.4.
3) The final proposed model, which uses DRBs in the decoding path, is denoted as ”Backbone-DRBs’’ in Table 5.4.

Method	*JAC*	*DIC*	*SEN*	*SPE*	*ACC*	*MCC*
Unet32 (baseline)	0.7524	0.8389	0.8470	0.9678	0.9290	0.8053
Backbone-CCBs	0.7673	0.8506	0.8476	0.9681	0.9335	0.8176
Backbone -DCBs	0.7783	**0.8605**	0.8642	**0.9715**	0.9362	**0.8279**
Backbone -DRBs	**0.7789**	0.8601	**0.8745**	0.9621	**0.9375**	**0.8262**

Bold values indicate the best results

Table 5.4. Ablation study for each contribution on the ISBI 2017 test dataset.

As shown in Table 5.4, the proposed model (Backbone-CCBs) provided better results than U-Net. This indicated that by using the PPM and DCB, the proposed model was able to further extract high-level features (semantic information) with high resolution when compared to U-Net. Introducing DCBs in the decoding path instead of CCBs (Backbone-DCBs) improved the overall performance. Using DRBs in the decoding path (Backbone-DRBs) further increased the Jaccard index (JAC), the sensitivity (SEN), and the accuracy (ACC), while decreased the dice coefficient (DIC), the specificity (SPE) and the (MCC). These results showed that using DCBs or DRBs in the decoding path helped to produce dense feature maps by extracting more context information, which boosted the segmentation performance. We adopted “Backbone-DRBs” as the final model regarding its best result in JAC index, which is the main metric of the ISBI 2017 and ISBI 2016 challenges.

B. Comparison with baselines and state-of-the-art methods

The quantitative results obtained are shown in Tables 5.5, 5.6, and 5.7. Our proposed model is compared to FCN, SegNet, U-Net, and U-Net++ as baseline models (under the same

conditions) and some latest existing methods. For U-Net, we started with 32 kernels (U-Net32) at the first layer of the network instead of 64 (U-Net64) in the original paper, since we noticed that U-Net32 performed better than U-Net64.

Reference	Year	*JAC*	*DIC*	*SEN*	*SPE*	*ACC*	*MCC*
DSNet [148]	2020	0.775	-	0.875	0.955	-	-
Res-Unet [152]	2020	0.772	0.858	-	-	-	-
iFCN [149]	2020	**0.7834**	**0.8864**	0.8544	0.9808	0.9530	-
Xie et al. [150]	2020	0.783	0.862	0.870	0.964	0.938	-
Pour et al. [151]	2020	0.782	0.871	**0.883**	0.981	0.945	-
CSARM-CNN [153]	2020	0.7335	0.8462	0.8022	**0.9940**	**0.9585**	0.8232
FCN (baseline)*		0.7267	0.8239	0.7943	0.9713	0.9275	0.7890
FCN (baseline)		0.7373	0.8320	0.8282	0.9664	0.9291	0.7960
SegNet (baseline)*		0.7646	0.8488	0.8266	0.9738	0.9341	0.8178
SegNet (baseline)		0.7704	0.8540	0.8593	0.9674	0.9352	0.8215
U-Net32(baseline)*		0.7457	0.8333	0.8138	0.9739	0.9276	0.8006
U-Net32(baseline)		0.7524	0.8389	0.8470	0.9678	0.9290	0.8053
U-Net++(baseline)*		0.7506	0.8353	0.8170	0.9765	0.9268	0.8046
U-Net++(baseline)		0.7498	0.8368	0.8459	0.9702	0.9277	0.8047
Model-10(Proposed) *		0.7775	0.8579	0.8441	0.9694	0.9370	0.8253
Model-10 (Proposed)		0.7789	0.8601	0.8745	0.9621	0.9375	**0.8262**

* Without any post-processing operations. Bold values indicate the best results

Table 5.5. Comparison of segmentation results on the ISBI 2017 test dataset.

Reference	Year	*JAC*	*DIC*	*SEN*	*SPE*	*ACC*	*MCC*
DCL-PSI [10]	2019	0.8592	0.9177	0.9311	0.9605	0.9578	-
Xie et al. [150]	2020	0.858	0.918	0.870	0.964	0.938	-
Pour et al. [151]	2020	0.852	0.921	**0.974**	0.949	0.961	-
FCN (baseline)*		0.8355	0.9045	0.9166	0.9490	0.9519	0.8637
SegNet (baseline)*		0.8421	0.9068	0.9226	0.9588	0.9530	0.8720
U-Net32(baseline)*		0.8345	0.9011	0.9394	0.9494	0.9512	0.8654
U-Net++(baseline)*		0.8403	0.9059	0.9239	0.9605	0.9540	0.8709
Model-10 (Proposed)*		**0.8639**	**0.9233**	0.9311	**0.9676**	**0.9621**	**0.8902**

* Without any post-processing operations. Bold values indicate the best results

Table 5.6. Comparison of segmentation results on the ISBI 2016 test dataset.

Reference	Year	*JAC*	*DIC*	*SEN*	*SPE*	*ACC*	*MCC*
DSNet [148]	2020	**0.870**	-	0.929	0.969	-	-
Res-Unet [152]	2020	0.854	0.924	-	-	-	-
iFCN [149]	2020	0.871	**0.9302**	0.9688	0.9531	**0.9692**	-
Xie et al. [150]	2020	0.857	0.919	0.963	0.942	0.949	-
CSARM-CNN [153]	2020	0.7909	0.8832	0.8854	**0.9945**	0.9523	0.8553
FCN (baseline)*		0.8131	0.8940	0.9585	0.9159	0.9321	0.8291
FCN (baseline)		0.8322	0.9056	0.9324	0.9383	0.9389	0.8451
SegNet (baseline)*		0.7987	0.8806	0.9531	0.9255	0.9273	0.8244
SegNet (baseline)		0.8199	0.8940	0.9269	0.9513	0.9349	0.8430
U-Net32(baseline)*		0.8183	0.8913	0.9443	0.9390	0.9287	0.8373
U-Net32(baseline)		0.8329	0.8999	0.9124	0.9630	0.9333	0.8511
U-Net++(baseline)*		0.8147	0.8931	0.9538	0.9387	0.9303	0.8364
U-Net++(baseline)		0.8393	0.9079	0.9277	0.9633	0.9368	0.8565
Model-10 (Proposed) *		0.8333	0.9052	**0.9701**	0.9272	0.9377	0.8497
Model-10 (Proposed)		0.8616	0.9222	0.9459	0.9551	0.9470	**0.8738**

* Without any post-processing operations. Bold values indicate the best results

Table 5.7. Comparison of segmentation results on the PH2 dataset.

Results on the ISBI 2017 test dataset

Table 5.5 shows the quantitative results on the ISBI 2017 test dataset (600 images). From the table, it can be observed that the proposed model (Model-10) achieved superior performance compared to the baseline models (FCN, SegNet, U-Net, and U-Net++). Furthermore, it achieved competitive results especially in terms of JAC (0.7789) and *SEN* (0.8745), when compared with latest state-of-the-art techniques.

Results on the ISBI 2016 test dataset

Table 5.6 displays the quantitative results on the ISBI 2016 test dataset (379 images). Because some state-of-the-art methods were not evaluated on this dataset, they are omitted in this table. The results show that the proposed model without any post-processing operations obtained superior segmentation performance in all metrics, except for *SEN*.

Results on the PH2 dataset

The quantitative results on the PH2 dataset (200 images) were compared against other methods, as shown in Table 5.7. The table shows that our model outperformed the baseline models and achieved state-of-the-art segmentation performance. It can be observed that the proposed Model-10 outperformed the latest state-of-the-art methods Res-Unet [152], Xie et al. [150], and CSARM-CNN [153] in terms of *JAC* (0.8616).

5.2.3.4 Discussion of the obtained results

To address the challenging task of automatic skin lesion segmentation, we proposed a simple encoder-decoder structure, which uses as the encoding path the VGG16 standard convolutional layers, followed by a PPM and a DCB, while uses DRBs in the decoding path. The number of VGG16 convolutional layers (in the encoder) is the key hyper-parameter in the proposed model because it controls the model parameters and segmentation performance. As shown in Table 5.3, the proposed model with 10 VGG16 convolutional layers (Model-10) performed better than Model-13 (with 13 layers) on both the PH2 dataset and the ISBI 2016 test dataset; and Model-7 on the PH2 and ISBI 2017 test datasets. Some segmentation results are displayed in Figs. 5.11 and 5.12. However, these figures show that both our model and others had the same performance on the normal examples (Fig. 5.11), while our model segmented better than others on the challenging examples, such as low contrast (Fig. 5.12).

Tables 5.5, 5.6, and 5.7 show that the proposed model performed better than baseline models (FCN, SegNet, U-Net, and U-Net++) on three public benchmark datasets under the same conditions. In terms of *JAC index*, our model achieved an increase of 4.16%, 0.85%, 2.65%, and 2.88% on the ISBI 2017 test dataset; an increase of 2.84%, 2.18%, 2.89%, and 2.36% on the ISBI 2016 test dataset; and an increase of 2.94%, 4.17%, 2.87%,and 2.23% on the PH2 dataset, when compared with FCN, SegNet, U-Net, and U-Net++, respectively. We attribute this improvement to the fact that the embedded PPM and DCB enable more and better representative of feature maps, while the DRBs further refine the prediction.

As shown in Tables 5.5, 5.6, and 5.7, the proposed model achieves state-of-the-art performance on the three public benchmark datasets ISBI 2017, ISBI 2016, and PH2. As it can be observed, our model provided better results than latest state-of-the-art techniques: DSNet [148], Res-Unet [152], and CSARM-CNN [153] in terms of *JAC*, and *DIC* metrics

on ISBI 2017 test dataset. On ISBI 2016 test dataset, it yielded the highest score in all metric except for *SEN*. On PH2 dataset, our network obtained higher performance than Res-Unet [152], Xie et al. [150] and CSARM-CNN [153] in terms of *JAC index*. Furthermore, the proposed model achieved competitive results in other metrics on three datasets. This stability of results indicates the robustness and consistency of the proposed method for skin lesion segmentation.

In addition, the computational time for both the training and testing stages is given in Table 5.8. As it can be seen, the processing time per epoch during the training stage of the proposed model was 106 s. This result was the third-best performance after U-Net and U-Net++. For the testing stage, like the model baselines, the computational time of our model took less than 2 s to process each image. Overall, compared to the baseline models (FCN, SegNet, U-Net, and U-Net++) and current state-of-the-art techniques (FrCN, iFCN, and DSNet), the proposed model is fast in both the training and testing stages. This fast processing time should make our model applicable for clinical practice.

Method	Year	Training time (sec.) per epoch	Test time (sec.) per image
FrCN [9]	2018	315	9.7
DSNet [148]	2020	-	0.595
iFCN [149]	2020	432.3	8
FCN		120	1.587
SegNet		119	1.356
U-Net		28	1.664
U-Net++		88	1.253
Model-10 (Proposed)		106	1.728

Table 5.8. Processing time during training and testing stages.

Although the proposed model has achieved satisfactory results on challenging cases (Fig. 5.12), there are some samples where the model had difficulty to correctly detect lesions, as shown in Fig. 5.13. It can be clearly seen that these examples are very challenging cases due to the very low contrast of lesions and their surrounding skin. For further improvement, the addition of other post-processing techniques, such as conditional random field (CRF) can enhance the model generalization.

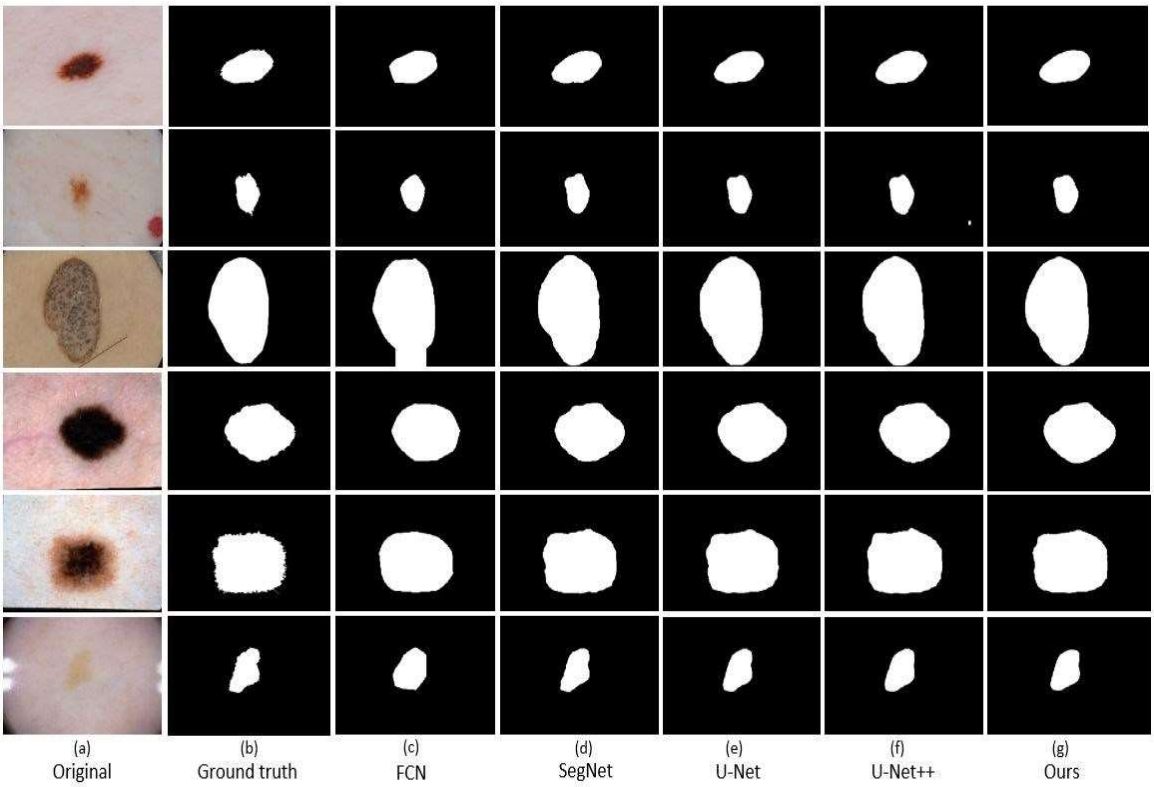

Figure 5.11. Segmentation results without difficulty of some normal samples of ISBI 2017 dataset.

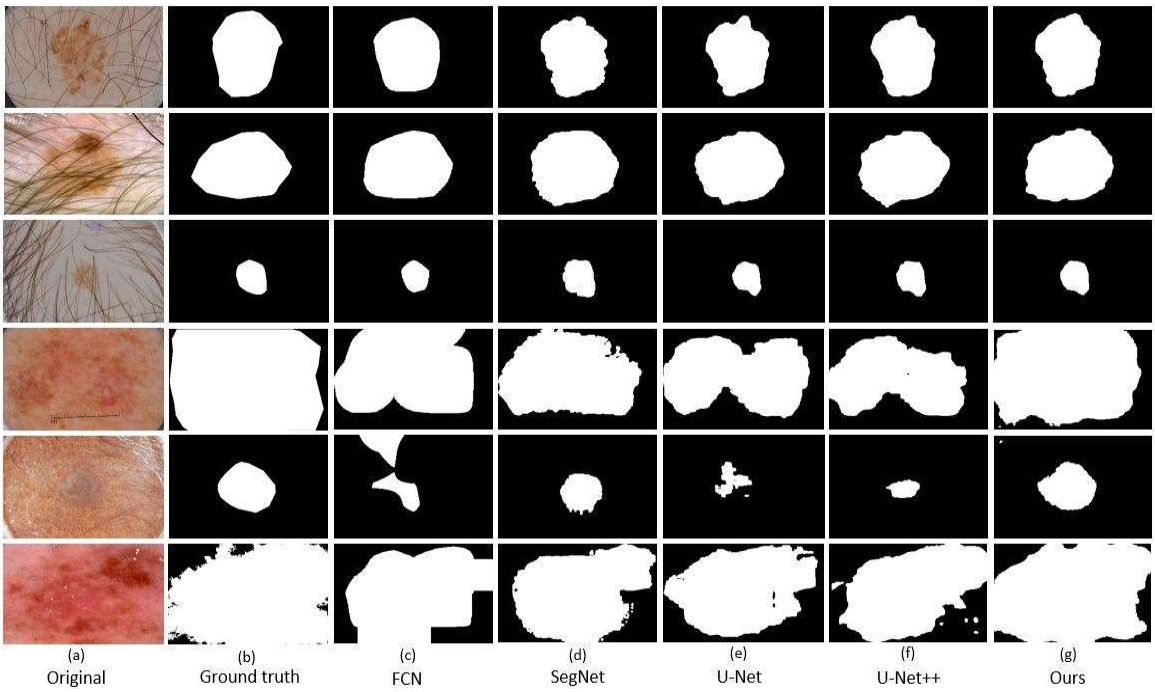

Figure 5.12 Segmentation results of some challenging samples of ISBI 2017 dataset.

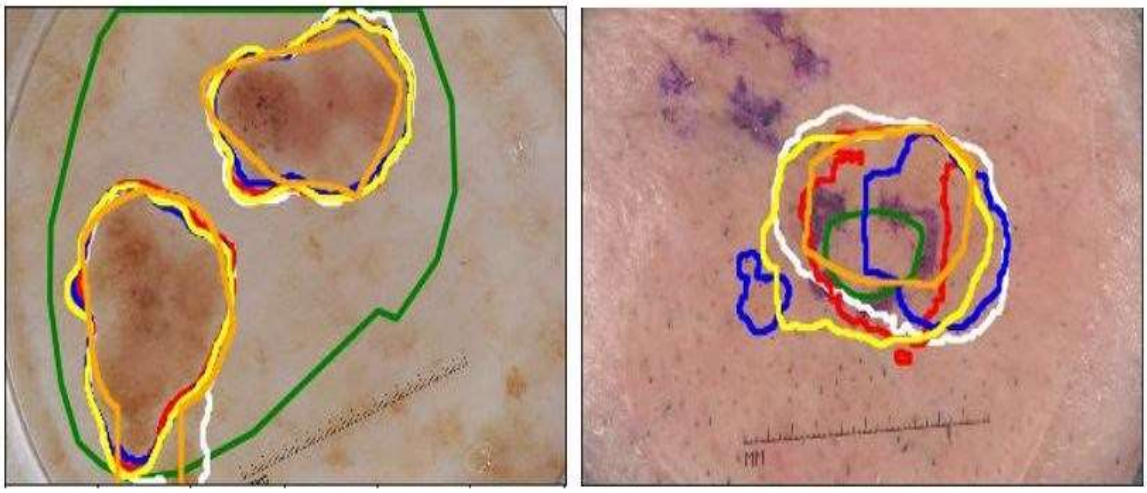

Figure 5.13. Some failure cases. (a) Under-segmentation, (b) over-segmentation. The green, red, orange, white, blue, and yellow contours represent the ground truth, the segmentation result of our method, FCN, SegNet, U-Net, and U-Net++ respectively.

5.2.3.5 Conclusion

Automatic skin lesion segmentation is a challenging and important task in the field of computer-aided decision systems. We proposed an encoder-decoder model based on the U-Net architecture with major modifications. The idea behind the encoding path is to integrate the pyramid pooling module (PPM) and dilated convolutional block (DCB) to the top of the first 10 convolutional layers of the VGG16 network. The PPM is used to capture information at multiple levels, while the DCB further maximizes features extraction ability with an enlarged receptive field. By using the proposed encoding path, our model was able to learn better representative features and outperformed U-Net on the very challenging ISBI 2017 dataset (Table 5.4). On the other hand, we introduced dilated residual blocks (DRBs) in the decoding path to replace the standard convolutional layers used in U-Net.

The DRBs, with their residual connections, facilitate the training process and further refine the segmentation maps. The experimental results showed that our model achieved the state-of-the-art performance on three public datasets, including ISBI 2017, ISBI 2016, and PH2. Although using a simple architecture with minimum pre- and post-processing operations, our method is robust to low contrast, and the presence of hair (Fig. 5.12). In addition, the proposed method is fast in both the training and testing phases.

Chapter 6
Conclusions

6.1 Conclusion

The incidence of skin cancer has been increasing in the world for the past decade. However, early diagnosis is essential for increasing survival chance and reducing mortality rate. Dermoscopy is one of the major tools in the early diagnosis. Generally, the manual analysis of dermoscopy images (visual interpretation) can be time-consuming, subjective and not reproducible. Therefore, computer aided diagnosis systems (CADs) can be used to minimize the diagnosis errors of manual analysis. Segmentation, an important step in CAD systems is a very active research area. This task is still a challenging issue due to the similarity between different lesions and complex visual characteristics that may be presented in the images. Recently, the state of the art techniques are based on deep learning, especially convolutional neural networks (CNNs). However, these methods require high computational time. In addition, some of these techniques use pre-processing and post-processing operations to obtain high performance. In this thesis, we addressed the problem of automatic skin lesion segmentation from dermoscopy images using CNNs. The proposed approaches are based on the state-of-the-art U-Net architecture. In the first approach, we used dilated convolution and pyramid pooling modules (PPM) to enhance the segmentation results. The second approach proposed a modified atrous spatial pyramid pooling (MASPP) module and integrated it as an input into each level of both the encoder and decoder. This strategy allows capturing multi-scale contextual information at each level of the network and thus leading for a better representative of feature maps. In the third approach (our main contribution), the proposed model adopted 10 standard convolutional layers followed by a pyramid pooling module (PPM) and a dilated convolutional block (DCB) as the encoding path, while dilated residual blocks (DRBs) were introduced in the decoding path. The results on several datasets, including ISBI 2017, ISBI 2016, and PH2 showed that this model outperformed U-Net, FCN, SegNet, and U-Net++, and achieved the performance of state-of-the-art segmentation techniques, with minimum pre- and post-processing operations. Furthermore, the proposed model is efficient in terms of computational time for both training and testing stages.

6.2 Future works

For future research directions, we will explore the results of a combination of three models: Backbone-CCBs, Backbone-DCBs, and Backbone-DRBs (from the Table 5.4); and model-7, model-10, and model-13 (from the Table 5.3) to produce the final segmentation maps. It would be interesting to conduct extensive experiments to study the impact of other loss functions and optimizers on the performance of the proposed models. Furthermore, we will integrate the classification task in the third approach (the main contribution), which will be explored on different color spaces, in addition to RGB color channels. Lastly, we will apply our proposed networks to other medical imaging datasets.

References

[1] C. Barata, M. E. Celebi, and J. S. Marques, "A Survey of Feature Extraction in Dermoscopy Image Analysis of Skin Cancer," *IEEE Journal of Biomedical and Health Informatics,* vol. 23, pp. 1096-1109, 2019.

[2] M. E. Celebi, N. Codella, and A. Halpern, " Dermoscopy Image Analysis: Overview and Future Directions," *IEEE Journal of Biomedical and Health Informatics* vol. V 23, pp. 474-478, 2019.

[3] https://www.cancer.org/cancer/melanoma-skin-cancer/about/key-statistics.html. accessed Fubruary. 22, 2023.

[4] M. E. Celebi, H. Iyatomi, G. Schaefer, and W. V. Stoecker, "Lesion border detection in dermoscopy images," *Computerized Medical Imaging and Graphics,* vol. 33, pp. 148-153, 2009.

[5] Y. Yuan, M. Chao, and Y.-C. Lo, "Automatic skin lesion segmentation using deep fully convolutional networks with jaccard distance," *IEEE Transactions on Medical Imaging,* vol. 36, pp. 1876-1886, 2017.

[6] R. Garnavi, M. Aldeen, M. E. Celebi, A. Bhuiyan, C. Dolianitis, and G. Varigos, "Skin Lesion Segmentation Using Color Channel Optimization and Clustering-based Histogram Thresholding," *International Journal of Biomedical and Biological Engineering,* vol. 36, pp. 365 - 373, 2009.

[7] M. Silveira, J. C. Nascimento, J. S. Marques, A. R. Marçal, T. Mendonça, S. Yamauchi, J. Maeda, and J. Rozeira, "Comparison of segmentation methods for melanoma diagnosis in dermoscopy images," *IEEE Journal of Selected Topics in Signal Processing,* vol. 3, pp. 35-45, 2009.

[8] M. E. Celebi, Q. Wen, H. Iyatomi, K. Shimizu, H. Zhou, and G. Schaefer, "A state-of-the-art survey on lesion border detection in dermoscopy images," *Dermoscopy image analysis,* vol. 10, pp. 97-129, 2015.

[9] M. A. Al-Masni, M. A. Al-Antari, M.-T. Choi, S.-M. Han, and T.-S. Kim, "Skin lesion segmentation in dermoscopy images via deep full resolution convolutional networks," *Computer methods and programs in biomedicine,* vol. 162, pp. 221-231, 2018.

[10] L. Bi, J. Kim, E. Ahn, A. Kumar, D. Feng, and M. Fulham, "Step-wise integration of deep class-specific learning for dermoscopic image segmentation," *Pattern Recognition,* vol. 85, pp. 78-89, 2019.

[11] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," presented at the CVPR, 2016.

[12] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. van der Laak, B. van Ginneken, and C. I. Sanchez, "A survey on deep learning in medical image analysis," *Med Image Anal,* vol. 42, pp. 60-88, Dec 2017.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp. 248-255.

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *: International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp. 234–241. Springer, Berlin (2015)*, 2015.

[15] B. Hafhouf, A. Zitouni, A. C. Megherbi, S. Sbaa, and . "A Modified U-Net for Skin Lesion Segmentation," presented at the 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), EL OUED, Algeria, pp. 225–228, IEEE (2020).

[16] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions," *CoRR,* vol. abs/1511.07122, 2016.

[17] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 6230-6239, 2017.

[18] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution,

and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 40, pp. 834-848, 2017.

[19] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587,* 2017.

[20] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801-818.

[21] B. Hafhouf, A. Zitouni, A. C. Megherbi, and S. Sbaa, "An Improved and Robust Encoder–Decoder for Skin Lesion Segmentation," *Arabian Journal for Science and Engineering,* pp. 1-15, 2022.

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

[23] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431-3440.

[24] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 39, pp. 2481-2495, 2017.

[25] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," In: 4th Deep Learning in Medical Image Analysis (DLMIA) and 8th Multimodal Learning for Clinical Decision Support (ML-CDS), pp. 3–11. Springer, Canada (2018)

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature,* vol. 521, pp. 436-444, 2015.

[27] A. Rosebrock, *Deep learning for computer vision with python: starter bundle*: PyImageSearch, 2017.

[28] T. M. Mitchell, "Machine learning," ed.

[29] Zhang, A.,Lipton, Z.C.,Li, M.,and Smola, A.J. (2021). Dive into deep learning. release 0.16.1,latest draft at http://d2l.ai/.

[30] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," T*he bulletin of mathematical biophysics, v*ol. 5, pp. 115-133, 1943.

[31] A. Karpathy, "Connecting images and natural language," Stanford University, 2016.

[32] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain," P*sychological review, v*ol. 65, p. 386, 1958.

[33] M. Minsky and S. Papert, "Perceptron: an introduction to computational geometry," ed: Cambridge, MA: MIT Press, 1969.

[34] D. E. Rumelhart, J. L. McClelland, and P. R. Group, "Parallel distributed processing: Explorations in the microstructure of cognition. Volume 1: Foundations'," ed: Cambridge, MA: MIT Press, 1986.

[35] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," N*ature, v*ol. 323, pp. 533-536, 1986.

[36] https://cs231n.github.io/optimization-1/

[37] https://medium.com/mlearning-ai/optimizers-in-deep-learning

[38] N. Qian, "On the momentum term in gradient descent learning algorithms," N*eural networks, v*ol. 12, pp. 145-151, 1999.

[39] https://www.willamette.edu/~gorr/classes/cs449/momrate.html

[40] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," J*ournal of machine learning research, v*ol. 12, 2011.

[41] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," C*OURSERA: Neural networks for machine learning, v*ol. 4, pp. 26-31, 2012.

[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," a*rXiv preprint arXiv:1412.6980, 2*014.

[43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," T*he journal of machine learning research, v*ol. 15, pp. 1929-1958, 2014.

[44] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," a*rXiv preprint arXiv:1502.03167, 2*015.

[45] F. Chollet, D*eep learning with Python:* Simon and Schuster, 2021.

[46] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, B. C. Van Esesn, A. A. S. Awwal, and V. K. Asari, "The history began from alexnet: A comprehensive survey on deep learning approaches," a*rXiv preprint arXiv: 1803.01164, 2*018

[47] I. Goodfellow, Y. Bengio, and A. Courville, D*eep learning:* MIT press, 2016.

[48] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, pp. 2278-2324, 1998.

[49] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of physiology,* vol. 160, p. 106, 1962.

[50] H. Li, Y. Pan, J. Zhao, and L. Zhang, "Skin disease diagnosis with deep learning: a review," *Neurocomputing,* vol. 464, pp. 364-393, 2021.

[51] https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp.1026-1034.

[53] A. Bhardwaj, W. Di, and J. Wei, *Deep Learning Essentials: Your hands-on guide to the fundamentals of deep learning and neural network modeling*: Packt Publishing Ltd, 2018.

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems,* vol. 25, 2012.

[55] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein, "Imagenet large scale visual recognition challenge," *International journal of computer vision,* vol. 115, pp. 211-252, 2015.

[56] A. Glassner, *Deep Learning: A Visual Approach*: No Starch Press, 2021.

[57] https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c

[58] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1-9.

[59] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818-2826.

[60] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[61] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251-1258.

[62] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

[63] D. Ciresan, A. Giusti, L. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," *Advances in neural information processing systems,* vol. 25, 2012.

[64] Z. Akkus, A. Galimzianova, A. Hoogi, D. L. Rubin, and B. J. Erickson, "Deep learning for brain MRI segmentation: state of the art and future directions," *Journal of digital imaging,* vol. 30, pp. 449-459, 2017.

[65] Li, F.-F., J. Johnson, and S. Yeung, Lecture 11: detection and segmentation, 2017, Stanford University http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf

[66] H. Noh, S. Hong, and B. Han, "Learning Deconvolution Network for Semantic Segmentation," presented at the Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), 2015.

[67] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 11-19.

[68] N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, "U-net and its variants for medical image segmentation: A review of theory and applications," *IEEE Access,* 2021.

[69] N. Ibtehaz and M. S. Rahman, "MultiResUNet: Rethinking the U-Net architecture for multimodal biomedical image segmentation," *Neural networks,* vol. 121, pp. 74-87, 2020.

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

[71] F. Yu, V. Koltun, and T. Funkhouser, "Dilated Residual Networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 636-644, 2017.

[72] F. Sultana, A. Sufian, and P. Dutta, "Evolution of image segmentation using deep convolutional neural network: a survey," *Knowledge-Based Systems,* vol. 201, p. 106062, 2020.

[73] S. Asgari Taghanaki, K. Abhishek, J. P. Cohen, J. Cohen-Adad, and G. Hamarneh, "Deep semantic segmentation of natural and medical images: a review," *Artificial Intelligence Review,* vol. 54, pp. 137-178, 2021.

[74] F. Lateef and Y. Ruichek, "Survey on semantic segmentation using deep learning techniques," *Neurocomputing,* vol. 338, pp. 321-348, 2019.

[75] X. Liu, L. Song, S. Liu, and Y. Zhang, "A review of deep-learning-based medical image segmentation methods," *Sustainability,* vol. 13, p. 1224, 2021.

[76] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2021.

[77] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, "A review on deep learning techniques applied to semantic segmentation," *arXiv preprint arXiv:1704.06857,* 2017.

[78] S. Ghosh, N. Das, I. Das, and U. Maulik, "Understanding deep learning techniques for image segmentation," *ACM Computing Surveys (CSUR),* vol. 52, pp. 1-35, 2019.

[79] S. Hao, Y. Zhou, and Y. Guo, "A brief survey on semantic segmentation with deep learning," *Neurocomputing,* vol. 406, pp. 302-321, 2020.

[80] I. R. I. Haque and J. Neubert, "Deep learning approaches to biomedical image segmentation," *Informatics in Medicine Unlocked,* vol. 18, p. 100297, 2020.

[81] M. Vestergaard, P. Macaskill, P. Holt, and S. Menzies, "Dermoscopy compared with naked eye examination for the diagnosis of primary melanoma: a meta-analysis of studies performed in a clinical setting," *British Journal of Dermatology,* vol. 159, pp. 669-676, 2008.

[82] F. Nachbar, W. Stolz, T. Merkle, A. B. Cognetta, T. Vogt, M. Landthaler, P. Bilek, O. Braun-Falco, and G. Plewig, "The ABCD rule of dermatoscopy: high prospective value in the diagnosis of doubtful melanocytic skin lesions," *Journal of the American Academy of Dermatology,* vol. 30, pp. 551-559, 1994.

[83] S. W. Menzies, C. Ingvar, K. A. Crotty, and W. H. McCarthy, "Frequency and morphologic characteristics of invasive melanomas lacking specific surface microscopic features," *Archives of dermatology,* vol. 132, pp. 1178-1182, 1996.

[84] G. Argenziano, G. Fabbrocini, P. Carli, V. De Giorgi, E. Sammarco, and M. Delfino, "Epiluminescence microscopy for the diagnosis of doubtful melanocytic skin lesions: comparison of the ABCD rule of dermatoscopy and a new 7-point checklist based on pattern analysis," *Archives of dermatology,* vol. 134, pp. 1563-1570, 1998.

[85] J. S. Henning, S. W. Dusza, S. Q. Wang, A. A. Marghoob, H. S. Rabinovitz, D. Polsky, and A. W. Kopf, "The CASH (color, architecture, symmetry, and homogeneity) algorithm for dermoscopy," *Journal of the American Academy of Dermatology,* vol. 56, pp. 45-52, 2007.

[86] H. Pehamberger, A. Steiner, and K. Wolff, "In vivo epiluminescence microscopy of pigmented skin lesions. I. Pattern analysis of pigmented skin lesions," *Journal of the American Academy of Dermatology,* vol. 17, pp. 571-583, 1987.

[87] https://upload.wikimedia.org/wikipedia/commons/c/c2/Dermatoscope.jpg

[88] Y. Gilaberte, L. Prieto-Torres, I. Pastushenko, and Á. Juarranz, "Anatomy and Function of the Skin," in *Nanoscience in Dermatology*, ed: Elsevier, 2016, pp. 1-14.

[89] G. Casey, "Physiology of the skin," *Nursing Standard (through 2013),* vol. 16, p. 47, 2002.

[90] C. Barata, M. E. Celebi, and J. S. Marques, "Explainable skin lesion diagnosis using taxonomies," *Pattern Recognition,* vol. 110, p. 107413, 2021.

[91] https://en.wikipedia.org/wiki/Skin

[92] https://www.cancer.org/cancer/basal-and-squamous-cell-skin-cancer.html

[93] https://www.skincancer.org/skin-cancer-information/basal-cell-carcinoma/

[94] https://www.skincancer.org/skin-cancer-information/squamous-cell-carcinoma/

[95] M. A. Kassem, K. M. Hosny, R. Damaševičius, and M. M. Eltoukhy, "Machine learning and deep learning methods for skin lesion classification and diagnosis: A systematic review," *Diagnostics,* vol. 11, p. 1390, 2021.

[96] J. Kawahara and G. Hamarneh, "Visual diagnosis of dermatological disorders: human and machine performance," *arXiv preprint arXiv:1906.01256,* 2019.

[97] T. Mendonça, P. M. Ferreira, J. S. Marques, A. R. Marcal, and J. Rozeira, "PH 2-A dermoscopic image database for research and benchmarking," in *2013 35th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, 2013, pp. 5437-5440.

[98] https://www.isic-archive.com/

[99] D. Gutman, N. C. Codella, E. Celebi, B. Helba, M. Marchetti, N. Mishra, and A. Halpern, "Skin lesion analysis toward melanoma detection: A challenge at the international symposium on biomedical imaging (ISBI) 2016, hosted by the international skin imaging collaboration (ISIC)," *arXiv preprint arXiv:1605.01397,* 2016.

[100] N. C. Codella, D. Gutman, M. E. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, A. Kalloo, K. Liopyris, N. Mishra, and H. Kittler, "Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic)," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, 2018, pp. 168-172.

[101] N. Codella, V. Rotemberg, P. Tschandl, M. E. Celebi, S. Dusza, D. Gutman, B. Helba, A. Kalloo, K. Liopyris, and M. Marchetti, "Skin lesion analysis toward

melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic)," *arXiv preprint arXiv:1902.03368,* 2019.

[102] P. Tschandl, C. Rosendahl, and H. Kittler, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific data,* vol. 5, pp. 1-9, 2018.

[103] A. Tharwat, "Classification assessment methods," *Applied Computing and Informatics,* 2020.

[104] M. Emre Celebi, Q. Wen, S. Hwang, H. Iyatomi, and G. Schaefer, "Lesion border detection in dermoscopy images using ensembles of thresholding methods," *Skin Research and Technology,* vol. 19, pp. e252-e258 %@ 0909-752X, 2013.

[105] L.-K. Huang and M.-J. J. Wang, "Image thresholding by minimizing the measures of fuzziness," *Pattern Recognition,* vol. 28, pp. 41-51, 1995.

[106] J. N. Kapur, P. K. Sahoo, and A. K. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram," *Computer vision, graphics, and image processing,* vol. 29, pp. 273-285, 1985.

[107] J. Kittler and J. Illingworth, "Minimum error thresholding," *Pattern Recognition,* vol. 19, pp. 41-47, 1986.

[108] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics,* vol. 9, pp. 62-66, 1979.

[109] M. E. Yüksel and M. Borlu, "Accurate segmentation of dermoscopic images by image thresholding based on type-2 fuzzy logic," *IEEE Transactions on Fuzzy Systems,* vol. 17, pp. 976-982, 2009.

[110] H. R. Tizhoosh, "Image thresholding using type II fuzzy sets," *Pattern Recognition,* vol. 38, pp. 2363-2372, 2005.

[111] S. Suer, S. Kockara, and M. Mete, "An improved border detection in dermoscopy images for density based clustering," in *BMC bioinformatics*, 2011, pp. 1-10.

[112] H. Zhou, G. Schaefer, A. H. Sadka, and M. E. Celebi, "Anisotropic mean shift based fuzzy c-means segmentation of dermoscopy images," *IEEE Journal of Selected Topics in Signal Processing,* vol. 3, pp. 26-34, 2009.

[113] H. Castillejos, V. Ponomaryov, L. Nino-de-Rivera, and V. Golikov, "Wavelet transform fuzzy algorithms for dermoscopic image segmentation," *Computational and mathematical methods in medicine,* vol. 2012, 2012.

[114] Q. Abbas, M. E. Celebi, I. Fondón García, and M. Rashid, "Lesion border detection in dermoscopy images using dynamic programming," *Skin Research and Technology,* vol. 17, pp. 91-100, 2011.

[115] H. G. Adelmann, "Butterworth equations for homomorphic filtering of images," *Computers in Biology and Medicine,* vol. 28, pp. 169-181, 1998.

[116] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on image processing,* vol. 13, pp. 1200-1212, 2004.

[117] Y. Yuan, M. L. Giger, H. Li, K. Suzuki, and C. Sennett, "A dual-stage method for lesion segmentation on digital mammograms," *Medical physics,* vol. 34, pp. 4180-4193, 2007.

[118] S. Lankton and A. Tannenbaum, "Localizing region-based active contours," *IEEE Transactions on image processing,* vol. 17, pp. 2029-2039, 2008.

[119] H. Iyatomi, H. Oka, M. E. Celebi, M. Hashimoto, M. Hagiwara, M. Tanaka, and K. Ogawa, "An improved internet-based melanoma screening system with dermatologist-like tumor area extraction algorithm," *Computerized Medical Imaging and Graphics,* vol. 32, pp. 566-579, 2008.

[120] M. Emre Celebi, H. A. Kingravi, H. Iyatomi, Y. Alp Aslandogan, W. V. Stoecker, R. H. Moss, J. M. Malters, J. M. Grichnik, A. A. Marghoob, and H. S. Rabinovitz, "Border detection in dermoscopy images using statistical region merging," *Skin Research and Technology,* vol. 14, pp. 347-353, 2008.

[121] R. Nock and F. Nielsen, "Statistical region merging," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 26, pp. 1452-1458, 2004.

[122] E. Ahn, J. Kim, L. Bi, A. Kumar, C. Li, M. Fulham, and D. D. Feng, "Saliency-based lesion segmentation via background detection in dermoscopic images," *IEEE journal of biomedical and health informatics,* vol. 21, pp. 1685-1693, 2017.

[123] L. Bi, J. Kim, E. Ahn, D. Feng, and M. Fulham, "Automated skin lesion segmentation via image-wise supervised learning and multi-scale superpixel based

cellular automata," in *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, 2016, pp. 1059-1062.

[124] A. Pennisi, D. D. Bloisi, D. Nardi, A. R. Giampetruzzi, C. Mondino, and A. Facchiano, "Skin lesion image segmentation using Delaunay Triangulation for melanoma detection," *Computerized Medical Imaging and Graphics,* vol. 52, pp. 89-103, 2016.

[125] B. Erkol, R. H. Moss, R. Joe Stanley, W. V. Stoecker, and E. Hvatum, "Automatic lesion boundary detection in dermoscopy images using gradient vector flow snakes," *Skin Research and Technology,* vol. 11, pp. 17-26, 2005.

[126] C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," *IEEE Transactions on image processing,* vol. 7, pp. 359-369, 1998.

[127] H. Zhou, G. Schaefer, M. E. Celebi, F. Lin, and T. Liu, "Gradient vector flow with mean shift for skin lesion segmentation," *Computerized Medical Imaging and Graphics,* vol. 35, pp. 121-127, 2011.

[128] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 17, pp. 790-799, 1995.

[129] Y. He and F. Xie, "Automatic skin lesion segmentation based on texture analysis and supervised learning," in *Asian Conference on Computer Vision*, 2012, pp. 330-341.

[130] A. R. Sadri, M. Zekri, S. Sadri, N. Gheissari, M. Mokhtari, and F. Kolahdouzan, "Segmentation of dermoscopy images using wavelet networks," *IEEE Transactions on Biomedical Engineering,* vol. 60, pp. 1134-1141, 2012.

[131] R. B. Oliveira, E. Mercedes Filho, Z. Ma, J. P. Papa, A. S. Pereira, and J. M. R. Tavares, "Computational methods for the image segmentation of pigmented skin lesions: a review," *Computer methods and programs in biomedicine,* vol. 131, pp. 127-141, 2016.

[132] K. Korotkov and R. Garcia, "Computerized analysis of pigmented skin lesions: a review," *Artificial intelligence in medicine,* vol. 56, pp. 69-90, 2012.

[133] S. Pathan, K. G. Prabhu, and P. Siddalingaswamy, "Techniques and algorithms for computer aided diagnosis of pigmented skin lesions—A review," *Biomedical Signal Processing and Control,* vol. 39, pp. 237-262, 2018.

[134] P. M. Pereira, R. Fonseca-Pinto, R. P. Paiva, P. A. Assuncao, L. M. Tavora, L. A. Thomaz, and S. M. Faria, "Dermoscopic skin lesion image segmentation based on local binary pattern clustering: comparative study," *Biomedical Signal Processing and Control,* vol. 59, p. 101924, 2020.

[135] J. L. Garcia-Arroyo and B. Garcia-Zapirain, "Segmentation of skin lesions in dermoscopy images using fuzzy classification of pixels and histogram thresholding," *Computer methods and programs in biomedicine,* vol. 168, pp. 11-19, 2019.

[136] F. F. X. Vasconcelos, A. G. Medeiros, S. A. Peixoto, and P. P. Reboucas Filho, "Automatic skin lesions segmentation based on a new morphological approach via geodesic active contour," *Cognitive Systems Research,* vol. 55, pp. 44-59, 2019.

[137] B. Bozorgtabar, M. Abedini, and R. Garnavi, "Sparse coding based skin lesion segmentation using dynamic rule-based refinement," in *international workshop on machine learning in medical imaging*, 2016, pp. 254-261.

[138] R. Kasmi, K. Mokrani, R. Rader, J. Cole, and W. Stoecker, "Biologically inspired skin lesion segmentation using a geodesic active contour technique," *Skin Research and Technology,* vol. 22, pp. 208-222, 2016.

[139] N. Moradi and N. Mahdavi-Amiri, "Kernel sparse representation based model for skin lesions segmentation and classification," *Computer methods and programs in biomedicine,* vol. 182, p. 105038, 2019.

[140] L. Yu, H. Chen, Q. Dou, J. Qin, and P.-A. Heng, "Automated Melanoma Recognition in Dermoscopy Images via Very Deep Residual Networks," *IEEE Transactions on Medical Imaging,* vol. 36, pp. 994-1004, 2017.

[141] L. Bi, J. Kim, E. Ahn, A. Kumar, M. Fulham, and D. Feng, "Dermoscopic Image Segmentation via Multistage Fully Convolutional Networks," *IEEE Transactions on Biomedical Engineering,* vol. 64, pp. 2065-2074, 2017.

[142] Y. Yuan and Y.-C. Lo, "Improving dermoscopic image segmentation with enhanced convolutional-deconvolutional networks," *IEEE journal of biomedical and health informatics,* vol. 23, pp. 519-526, 2017.

[143] Z. Mirikharaji, S. Izadi, J. Kawahara, and G. Hamarneh, "Deep auto-context fully convolutional neural network for skin lesion segmentation," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, 2018, pp. 877-880.

[144] P. Tang, Q. Liang, X. Yan, S. Xiang, W. Sun, D. Zhang, and G. Coppola, "Efficient skin lesion segmentation using separable-Unet with stochastic weight averaging," *Computer methods and programs in biomedicine,* vol. 178, pp. 289-301, 2019.

[145] M. Sarker, M. Kamal, H. A. Rashwan, F. Akram, S. F. Banu, A. Saleh, V. K. Singh, F. U. Chowdhury, S. Abdulwahab, and S. Romani, "SLSDeep: Skin lesion segmentation based on dilated residual and pyramid pooling networks," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2018, pp. 21-29.

[146] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International journal of computer vision,* vol. 92, pp. 1-31, 2011.

[147] A. H. Shahin, K. Amer, and M. A. Elattar, "Deep convolutional encoder-decoders with aggregated multi-resolution skip connections for skin lesion segmentation," in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, 2019, pp. 451-454.

[148] M. K. Hasan, L. Dahal, P. N. Samarakoon, F. I. Tushar, and R. Martí, "DSNet: Automatic dermoscopic skin lesion segmentation," *Computers in Biology and Medicine,* vol. 120, p. 103738, 2020.

[149] Ş. Öztürk and U. Özkaya, "Skin lesion segmentation with improved convolutional neural network," *Journal of digital imaging,* vol. 33, pp. 958-970, 2020.

[150] F. Xie, J. Yang, J. Liu, Z. Jiang, Y. Zheng, and Y. Wang, "Skin lesion segmentation using high-resolution convolutional neural network," *Computer methods and programs in biomedicine,* vol. 186, p. 105241, 2020.

[151] M. P. Pour and H. Seker, "Transform domain representation-driven convolutional neural networks for skin lesion segmentation," *Expert Systems with Applications,* vol. 144, p. 113129, 2020.

[152] K. Zafar, S. O. Gilani, A. Waris, A. Ahmed, M. Jamil, M. N. Khan, and A. Sohail Kashif, "Skin lesion segmentation from dermoscopic images using convolutional neural network," *Sensors,* vol. 20, p. 1601, 2020.

[153] Y. Jiang, S. Cao, S. Tao, and H. Zhang, "Skin lesion segmentation based on multi-scale attention convolutional neural network," *IEEE Access,* vol. 8, pp. 122811-122825, 2020.

[154] S. Khouloud, M. Ahlem, T. Fadel, and S. Amel, "W-net and inception residual network for skin lesion segmentation and classification," *Applied Intelligence,* vol. 52, pp. 3976-3994, 2022.

[155] M. Goyal, A. Oakley, P. Bansal, D. Dancey, and M. H. Yap, "Skin lesion segmentation in dermoscopic images with ensemble deep learning methods," *IEEE Access,* vol. 8, pp. 4171-4181, 2019.

[156] E. Nasr-Esfahani, S. Rafiei, M. H. Jafari, N. Karimi, J. S. Wrobel, S. Samavi, and S. R. Soroushmehr, "Dense pooling layers in fully convolutional network for skin lesion segmentation," *Computerized Medical Imaging and Graphics,* vol. 78, p. 101658, 2019.

[157] L. Zhang, G. Yang, and X. Ye, "Automatic skin lesion segmentation by coupling deep fully convolutional networks and shallow network with textons," *Journal of Medical Imaging,* vol. 6, p. 024001, 2019.

[158] L. Liu, Y. Y. Tsui, and M. Mandal, "Skin lesion segmentation using deep learning with auxiliary task," *Journal of Imaging,* vol. 7, p. 67, 2021.

[159] S. Qamar, P. Ahmad, and L. Shen, "Dense encoder-decoder–based architecture for skin lesion segmentation," *Cognitive Computation,* vol. 13, pp. 583-594, 2021.

[160] P. Tschandl, C. Sinz, and H. Kittler, "Domain-specific classification-pretrained fully convolutional network encoders for skin lesion segmentation," *Computers in Biology and Medicine,* vol. 104, pp. 111-116, 2019.

[161] Y. Ren, L. Yu, S. Tian, J. Cheng, Z. Guo, and Y. Zhang, "Serial attention network for skin lesion segmentation," *Journal of Ambient Intelligence and Humanized Computing,* vol. 13, pp. 799-810, 2022.

[162] S. Baghersalimi, B. Bozorgtabar, P. Schmid-Saugeon, H. K. Ekenel, and J.-P. Thiran, "DermoNet: densely linked convolutional neural network for efficient skin

lesion segmentation," *EURASIP Journal on Image and Video Processing,* vol. 2019, pp. 1-10, 2019.

[163] A. A. Adegun, S. Viriri, and M. H. Yousaf, "A Probabilistic-Based Deep Learning Model for Skin Lesion Segmentation," *Applied Sciences,* vol. 11, p. 3025, 2021.

[164] H. Li, X. He, F. Zhou, Z. Yu, D. Ni, S. Chen, T. Wang, and B. Lei, "Dense deconvolutional network for skin lesion segmentation," *IEEE journal of biomedical and health informatics,* vol. 23, pp. 527-537, 2018.

[165] K. Sanjar, O. Bekhzod, J. Kim, J. Kim, A. Paul, and J. Kim, "Improved U-net: fully convolutional network model for skin-lesion segmentation," *Applied Sciences,* vol. 10, p. 3658, 2020.

[166] Y. Dong, L. Wang, S. Cheng, and Y. Li, "Fac-net: Feedback attention network based on context encoder network for skin lesion segmentation," *Sensors,* vol. 21, p. 5172, 2021.

[167] G. Arora, A. K. Dubey, Z. A. Jaffery, and A. Rocha, "Architecture of an effective convolutional deep neural network for segmentation of skin lesion in dermoscopic images," *Expert Systems,* p. e12689, 2021.

[168] L. Huang, Y.-g. Zhao, and T.-j. Yang, "Skin lesion segmentation using object scale-oriented fully convolutional neural networks," *Signal, Image and Video Processing,* vol. 13, pp. 431-438, 2019.

[169] M. A. Al-Masni and D.-H. Kim, "CMM-Net: contextual multi-scale multi-level network for efficient biomedical image segmentation," *Scientific reports,* vol. 11, pp. 1-18, 2021.

[170] L. Huang, Y.-g. Zhao, and T.-j. Yang, "Skin lesion segmentation using object scale-oriented fully convolutional neural networks," *Signal, Image and Video Processing,* vol. 13, pp. 431-438, 2019.

[171] M. Rahman, N. Alpaslan, and P. Bhattacharya, "Developing a retrieval based diagnostic aid for automated melanoma recognition of dermoscopic images," in *2016 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, 2016, pp. 1-7.

[172] S. S. M. Salehi, D. Erdogmus, and A. Gholipour, "Tversky loss function for image segmentation using 3D fully convolutional deep networks," in *International workshop on machine learning in medical imaging*, 2017, pp. 379-387.

[173] S. R. Hashemi, S. S. M. Salehi, D. Erdogmus, S. P. Prabhu, S. K. Warfield, and A. Gholipour, "Tversky as a loss function for highly unbalanced image segmentation using 3d fully convolutional deep networks," *arXiv preprint arXiv:1803.11078,* 2018.

[174] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, "Understanding convolution for semantic segmentation. arXiv e-print," *arXiv preprint arXiv:1702.08502,* 2017.

[175] J. Dolz, X. Xu, J. Rony, J. Yuan, Y. Liu, E. Granger, C. Desrosiers, X. Zhang, I. Ben Ayed, and H. Lu, "Multiregion segmentation of bladder cancer structures in MRI with progressive dilated convolutional networks," *Medical physics,* vol. 45, pp. 5482-5493, 2018.

[176] Z. Wang and S. Ji, "Smoothed dilated convolutions for improved dense prediction," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2486-2495.