Dense Fully Convolutional Network for Skin Lesion Segmentation

Ebrahim Nasr-Esfahani, Shima Rafiei, Mohammad H. Jafari, Nader Karimi, James S. Wrobel, S.M. Reza Soroushmehr, Shadrokh Samavi, Kayvan Najarian

Abstract—Skin cancer is a deadly disease and is on the rise in the world. Computerized diagnosis of skin cancer can accelerate the detection of this type of cancer that is a key point in increasing the survival rate of patients. Lesion segmentation in skin images is an important step in computerized detection of the skin cancer. Existing methods for this aim usually lack accuracy especially in fuzzy borders of lesions. In this paper, we propose a new class of fully convolutional networks with novel dense pooling layers for segmentation of lesion regions in non-dermoscopic images. Unlike other existing convolutional networks, the proposed dense pooling layers are designed to preserve all of the input features. This has led to highly accurate segmentation of lesions. Our proposed method produces dice score of 91.6% which outperforms all state-of-the-art algorithms in segmentation of skin lesions based on the Dermquest dataset.

Index Terms—skin cancer, melanoma, deep neural networks, dense pooling layer.

I. INTRODUCTION

Comptuerized diagnosis of skin cancer is of great need and interest [1]. About 5.4 million new cases of skin cancer are detected in the USA each year. Most fatal types of skin cancer are melanoma, which cause 75% related deaths [2]. Recently the incidence pattern of melanoma has shown a rapid increase. Rate of melanoma occurrence is tripled in the past 30 years [3]. In the USA, an estimated of 87,110 new detected cases and an estimated 9,730 melanoma related deaths occur in 2017. A key point in survival of patients is early detection of malignant skin lesions [4-5]. Patients, with melanoma detected in the early stages, have a 98% of five-year relative chance of survival. The survival rate is shown to be only 18% when melanoma is spread to the other parts of the body, which results in a life expectancy with median of less than one year [2-3].

The importance and variety of computerized methods for early detection of melanoma are reviewed in [6-7]. There are two main categories of images used for computerized diagnosis of melanoma. One category belongs to Dermoscopic images, also known as microscopic images, which are captured by a special purpose instrument called dermoscope. The second category is non-dermoscopic images, which are images captured by conventional digital cameras and smartphones. Dermoscopic images have more detailed information but non-dermoscopic images have the advantage of ease of access. Dermoscopic images are not easily accessible. In [8] it is indicated that less than 50% of the dermatologists in the United States use dermatoscopy. In this paper we consider methods using non-dermoscopic images, which are also called macroscopic or clinical images [9].

One of the most important stages in computerized study of melanoma is accurate segmentation of skin lesions. Captured images usually contain a lesion, which is surrounded by normal skin. Accurate extraction of the lesion region is critical for assessment of lesion’s features. Lesion features include area, border irregularity, shape symmetry, and variation of color within the lesion. Various methods exist for skin lesions segmentation, based on active contours, region merging [10], and thresholding [11]. Non-dermoscopic images show what is seen by the naked eye. There are some challenges in segmentation of non-dermoscopic image, such as presence of fuzzy borders between normal skin and lesion, variety of textures and colors of lesions, illumination variations in captured images, and presence of skin artifacts, such as hair. Some researchers have proposed methods to specifically tackle issues existed in segmentation of non-dermoscopic images [12-16].

Methods of [12], [13], and [14] are based on typical image processing techniques, which consider vectors of colors and texture features to classify lesion pixels. In [12] the red channel of RGB images is considered to discriminate the lesion from background. In [13], to enhance the segmentation result, it models the illumination variation by quadratic function and relights the skin pixels to degrade the shading effects in the RGB image. The algorithm of [14] generates a discriminative representation of a 3-channel image to distinguish between lesion and background.
These methods usually lack accuracy. Methods of [15], learned sparse texture distributions of the lesion and normal images are considered to discriminate between normal and lesion regions. This approach resulted in noticeable improvement, but it performs poorly in some cases when complex lesion and skin patterns are present.

A relatively successful and widely used method is deep learning which has shown an emerging growth in different fields of computer vision [17]. In particular, deep convolutional neural networks have resulted in major breakthroughs in different aspects of medical image analysis [18]. Currently deep learning methods are the state-of-the-art framework for various medical imaging challenges, such as classification of mammographic lesions [19], brain lesion segmentation [20], detection of hemorrhage in color fundus images [21], and bone suppression in X-ray images [22].

Method of [16] is a recent approach presented for segmentation of non-dermoscopic images. In [16] deep convolutional neural networks are applied for lesion segmentation. Work of [16] applies a deep network involving information of local and global windows around regions of the image. Network of [16] classifies pixels into one of the two groups of lesion, and normal skin, by employing both local and global views. Method of [16] is based on sliding-window, which makes it very time consuming despite the small structure of its network. Moreover in some lesion cases it suffers from the lack of accuracy.

In this paper, we propose a new class of network, which we call Dense Fully Convolutional Network (DFCN). The proposed network generates high resolution dense feature maps with no need for upsampling similar to sliding-window based methods. These feature maps are generated almost as fast as the fully convolutional networks (FCNs). Hence, the proposed approach has the dense resolution of sliding-window based methods, and the speed of fully convolutional networks.

We have applied this proposed network for lesion segmentation and we have achieved higher dice scores as compared to other state of the art methods. Briefly, our innovations are

- Proposal of a new class of fully convolutional networks with high accuracy and high speed.
- Accurate segmentation of skin lesions for detection of melanoma.

The rest of the paper is organized such that in Section II typical data feeding methods for convolutional neural networks are presented. Then in Section III we present our novel proposed architecture for dense fully convolutional network. Proposed lesion segmentation method is presented in Section IV and experimental results are shown in Section V. Concluding remarks are offered in Section VI.

II. TYPICAL DATA FEEDING IN CONVOLUTIONAL NEURAL NETWORKS

Typical convolutional neural networks (CNNs) are made up of a series of consecutive convolution and pooling layers. The input to the first layer of the CNN is an original image. In each convolutional layer, the input is convolved with a predetermined number of kernels to produce feature maps. Moreover, by passing through each pooling layer, the sizes of the maps are reduced according to the kernel and the stride size of the pooling layers.

For image segmentation, a number of end-to-end networks are proposed. An image is fed into the network and many features are obtained. This is done to eventually achieve a probability map. This map will show the probability of each pixel’s membership to each class. In CNN structures, to receive an image as an input, two different methods are now being used. One method is to have a sliding-window where each pixel of the image is sent to the network using a window around that pixel. The second method is to send the whole image to a fully convolutional network. To illustrate how different methods process the input data we use a one-dimensional example. Suppose we want to process a 1-D data string with 15 elements of D1 to D15. In the following two sub-sections, using this 1-D example, we consider two convolutional networks. For the illustration purpose, each of the two convolutional networks has two convolution layers and one pooling layer. In the following the two inputting methods are explained using the mentioned example.

A. Sliding-window method

In the sliding-window method, to determine the class of each pixel, a window is considered around the pixel. The window is fed into the network where it goes through the convolutional and pooling layers. This will result in extraction of features. The extracted features are finally applied to a fully connected network, which by using a softmax layer, would output a probability for the central pixel of the window. The probability production procedure is repeated for all pixels of the input image. This is done by sliding the window on all pixels. By obtaining probabilities for all pixels and by applying a thresholding process on the obtained probability map, a classification map for the input image is resulted.

If a small size window is fed to network, a local view around the pixels will be considered that contains local texture and the precise edge locations. Otherwise, if a large window is defined then a global view is achieved. Global view could extract some general structures and could be beneficial for some applications and could facilitate the classification process. An efficient approach is to combine local and global information by processing both local and global views, as used in [16] and [23].

Sliding-window methods have the capability of being used in problems where the size of the training data is limited. In window-based training, each pixel in the image, along with its surrounding pixels in the window, can be regarded as a training sample. Hence, the number of training elements is almost equal to the number of pixels in the training images. The main issue here would be the existing redundancy in calculations that are performed by the network. Since adjacent windows have a considerable number of overlapping pixels, the convolutional and pooling operations, to assign labels to adjacent pixels, would have major overlaps. Another drawback is that the label assignment for each pixel is done...
separately; therefore, the valuable information about the labels of the surrounding pixels is not involved in the classification procedure. This isolated classification in sliding-window methods could result in poor accuracy in some applications.

As shown in Fig. 1, a sliding-window with the size of 8, is slid on the input data. Eventually 8 features of F1 to F8 are obtained. In other words, a single feature is obtained from its corresponding receptive field, or input window of size 8.

B. Fully convolutional method

Since in the sliding-window based methods, the network is terminated by fully connected layers, the size of the input window must be fixed. In fully convolutional network (FCN) by replacing the fully connected layers with a series of deconvolution layers, the size of the input image could vary and it no longer depends on the network’s structure.

The general structure of FCN comprises of two main phases. It consists of convolutional encoder and convolutional decoder. In FCN, contrary to the sliding-window approach, the image as a whole is fed to the network. At the convolutional encoder phase, the entire image is passed through convolutional and pooling layers. As a result, low resolution feature maps are obtained. The convolutional decoder deconvolves the produced low resolution feature maps into a map in almost the same size as the input image. Hence, the network generates probabilities for each pixel. Deconvolution is usually done by one step interpolation [24] or by a series of deconvolution and upsampling layers [25-26].

The accuracy of the FCN is highly dependent on the accurate deconvolution of the low resolution feature maps. In [26] and [27], the indices of pooled features, in pooling layers of the convolutional encoder, are stored. The stored locations and their indices are then used in the convolutional decoder phase to improve the accuracy for generating the final map. In [25] a U-shape network structure for medical image segmentation is proposed. In each deconvolution layer of their network, the corresponding feature maps from the encoder stage are concatenated with the upsampled maps in order to improve the deconvolution accuracy.

By applying the FCN method, as can be seen in Fig. 2, the entire input data is convolved in one step. Then after pooling and second convolutional layer, 4 features, namely F1, F3, F5, and F7 are obtained. Actually, each output feature corresponds with a specific input window. However, some output features, such as F2, F4, F6, and F8 are not generated due to the presence of the max pooling layer with stride 2.

The pooling layer, with stride 2, would ignore some of the overlapped input windows. As a result, the corresponding output features would be missing. In the next step, the missing features must be reconstructed by deconvolution layers, which may rebuild the missing outputs imprecisely. It should be noted that various methods are proposed to produce better estimation of the missing features using complex upsampling algorithms.

Compared to the sliding-window method, in FCN by reducing the redundancy in the network’s operations for adjacent pixels, the network’s training and testing speeds are increased considerably. Meanwhile, the added deconvolution decoder layers increases the number of required parameters and the required calculations, which could impair the network’s learning process.

III. DENSE FULLY CONVOLUTIONAL NETWORK

The proposed Dense Fully Convolutional Network (DFCN) is explained in this section. In DFCN, by introducing an innovative pooling method, the computational redundancies in calculations of adjacent windows in the sliding-window approaches are avoided. The DFCN’s encoder directly outputs high resolution feature maps with almost the same size as the input image. Therefore, in DFCN the need for the
deconvolution phase is eliminated. The absence of the deconvolution phase causes DFCN to have high speed that is a characteristic of FCNs. On the other hand, DFCN uses an innovative pooling method that eliminates the deconvolution steps in FCNs.

In FCNs, a pooling layer divides a 2-D input into a set of non-overlapping rectangular sub-regions, which are commonly 2×2. The pooling layer performs a non-linear function, which is commonly a max pooling operation with a stride of 2. This means the pooling layer reduces the spatial size of the feature maps for the subsequent layers. Hence, max pooling strongly reduces the number of parameters for the rest of the network. Moreover each neuron in the subsequent layer, which follows the pooling layer, would have larger field-of-view. This larger field of view, which causes reception of information from a larger neighborhood of pixels, is resulted by the pooling effect. Hence, pooling would lead to a more powerful detection process. Nevertheless, this approach causes missing of some features and production of low resolution maps, which need reconstruction by deconvolution layers.

In DFCN, instead of using non-overlapping rectangular sub-regions, the pooling layers consider all possible overlapped sub-regions. This is done such that the missing information is preserved.

To illustrate how DFCN works, suppose we have a simple 1-D network as shown in Fig. 3. Similar to the networks of figures 1 and 2, this network has two convolution layers, both with kernels of size 3, and a pooling layer, with a kernel of size 2 and stride of 2. Unlike the networks of figures 1 and 2, the pooling layer in Fig. 3 is a dense-pooling layer. The shown network is one-dimensional, while it can easily be generalized to higher dimensional spaces.

As shown in Fig. 3, the input data, after convolution in the first layer, would be fed into the dense-pooling layer. In the dense pooling layer, at first, there would be a pooling with stride of 1. Then odd and even features are separated in two upper and lower categories. The upper category includes the features that would be produced by the FCN of Fig. 2, and the lower category includes the features that are ignored by the FCN.

Afterward, both categories are fed into the next convolutional layer and would be convolved with the same kernel using sharing weights. At the final stage of the network, all features that are in different categories are interlaced and they are placed in their correct locations. As can be seen, the DFCN’s output would be exactly the same as the sliding-window method due to the fact that the dense pooling layer do not ignore any receptive field. As a result, DFCN produces a high resolution feature map without the need for any deconvolution step. Hence, large number of parameters are eliminated, which results in a network that is easier to train.

On the other hand, DFCN produces the high resolution feature maps of sliding-window based networks and high speed of FCNs. The speedup of DFCN is achieved because all of overlapping computations, which exist in the sliding-window methods, are avoided by dense pooling.

In all of the above mentioned networks, the structures and the number of parameters are the same, while the number of operations required to produce these features are different. In Table I the number of multiplications and number of additions are shown for each network. The number of features achieved by sliding-window is identical to that of DFCN. Nevertheless, since we removed redundancies in calculations, the number of operations used by the sliding-window is almost 3 times more than DFCN. In addition, the number of operations in DFCN is almost 20 percent more than the number of operations in FCN. However the features produced by DFCN are more precise and there are no missing feature. Hence, unlike FCN, there is no need for additional deconvolution layers for the reconstruction of the missing features.

| Method       | # of Multiplications | # of Additions |
|--------------|----------------------|----------------|
| Sliding-Window | 168                  | 72             |
| FCN          | 51                   | 19             |
| DFCN         | 63                   | 23             |

To generalize, when a 2-dimentional image is fed into the DFCN, the first dense pooling layer, with stride $S$, at the first step, pools each of its input feature maps with stride one. Then it splits set of one-stride maps. This means that each single category is split into $S^2$ categories. All these generated categories are of equal sizes with no overlap. Also, the union of these categories is the same as the set of one-stride maps. In the same way, each subsequent dense pooling layer would split each of its input categories into $S^2$ categories. As a result,
the number of categories is increased over the network.

The splitting procedure is explained by (1). In (1), for each category, each feature map, \( F_i \), is split and placed into \( S^2 \) categories. In (1), \( m \) and \( n \) are the row and column indices of the new categories, while \( x \) and \( y \) indicate the spatial location in each category.

\[
F^{mn}(x/s, y/s) = F(x/s + m, y/s + n); \quad \forall (m, n) \in \{0, 1, \ldots, S - 1\}
\]  

IV. PROPOSED LESION SEGMENTATION METHOD

We have developed a network based on DFCN structure for segmentation of skin lesions in non-dermoscopic images. In this section, the structure of network is described, followed by the strategies applied for improving the training process and increasing the effectiveness of the network.

A. Network architecture

The DFCN structure of Fig. 4 converts an RGB image into a posterior probability map, which will eventually result in the segmentation of the image. The input that is fed into DFCN may be the whole image or a part of the image. The output of DFCN is a probability map, indicating the probability of each pixel’s membership to the lesion or the background region. The proposed DFCN only consists of convolutional and dense pooling layers.

In the proposed DFCN, all dense pooling layers are max pooling with 2x2 kernels and stride 2. Hence, each dense pooling would split each category that exists in the previous layer into 4 categories. For this aim, at the first step each dense pooling layer would pool every category with stride one according to (2). In this equation, for each category, variable \( F_i \) is the \( j \)th feature map of the \( l \)th layer. At the second step, since we are using stride 2, the dense pooling would split each pooled category into 4 categories.

\[
F^j_l(x,y) = \max \{F^{l-1}_{i} (x + m, y + n); \forall (m, n) \in \{0, 1\} \}
\]  

At the convolutional layer each category would be convolved according to (3), where: \( N^l \) is the number of feature maps of the category in the \( l \)th layer. Also, \( w_{ij}^l \) is the \( ij \)th kernel applied to the \( j \)th feature map of the category and \( b_i^l \) is the \( i \)th bias for this category.

It should be noted that all convolutional kernels are shared among all categories that exist in each layer.

\[
F^l_i = \sum_{l=1}^{N^l} (F^{l-1}_i * w^l_i) + b_i^l 1_{N^l}
\]  

Generally, the primary layers of the network contain low-level information such as border locations. On the other hand, as we traverse through the network, later layers contain high-level information such as the structural shapes. Hence, the network combines both high and low levels of information, leading to further precision in the segmentation result. For this aim, after cropping and convolving feature maps that are obtained from various layers, subsequent interlace layers would relocate each feature to its correct location. This leads to set of high resolution feature maps. Then the network concatenates these dense feature maps, which will be followed by two convolution layers. At the final stage of the network, the corresponding probability map would be produced, using softmax function of (4).

\[
p_k(x, y) = \exp(a_k(x, y)) / \sum_{k'=1}^{C} \exp(a_{k'}(x, y))
\]  

In (4), \( a_k(x, y) \) and \( p_k(x, y) \) are respectively the \( k \)th activation output and its probability of belonging to the \( k \)th class at the location \((x, y)\). Also, \( C \) is the number of classes. For lesion segmentation \( C \) is 2, which means that there are two classes of lesion and background.

B. DFCN training

The number of non-dermoscopic images in the training set as compared to the number of network parameters, is very small. Thus to enhance DFCN performance and overcome a possible overfitting problem derived from lack of the training set, we implemented five strategies that are explained in the following.

1) Sub-Imaging

Similar to FCN, our DFCN does not contain any fully connected layer. Hence, in the test or training phases, the entire or a part of an image with different sizes can be fed into the network. In our approach, the network is trained with sub-images of the size 155x155, which are cropped from the original image.

Since a major portion of a non-dermoscopic image belongs to the background area, pure random selection of the sub-images would cause an unbalanced training set. This unbalance training set leads to reduction of the performance of the network. Therefore, we select sub-images such that 25% of the sub-image centers are located in the lesion area and 25% are in the background area. In addition, 50% of the sub-image centers are within the lesion border areas, which are computed using morphological procedure of (5).

\[
border = (GT \circ SE) - (GT \oplus SE)
\]  

where \( \circ \) and \( \oplus \) are the dilation and erosion operators, respectively. Also, the \( GT \) shows the segmentation ground truth while \( SE \) is a disk structuring-element of size 20.

2) Pixel weighting

The last layer of the network is the soft-max layer, where the probability of each pixel’s membership to the lesion or background areas is determined. Our DFCN has been trained by the cross entropy loss function that is shown in (6).

\[
E = - \sum_{(x,y) \in Z^2} w(x,y) \log(p_c(x,y))
\]
where $p_c(x, y)$ is the probability of the pixel in location $(x, y)$ belonging to the lesion area, which is obtained from the softmax function. Also, $w(x, y)$ is a 2-D weighting map that is calculated by (7). Using (7), pixels that are closer to the boundary would gain higher weights. In other words, $w(x, y)$ would increase the importance of pixels that are closer to the lesion boundaries. Labeling such pixels is relatively hard since borders are usually vague.

In (7), $d(x, y)$ is the distance between the pixel and the lesion boundary. In this case, $w_0$ and $\sigma$ are selected to be 20 and 30, respectively.

$$w(x, y) = 1 + w_0 \cdot \exp\left(-\frac{d(x, y)}{2\sigma^2}\right)$$  \hspace{1cm} (7)

### 3) Adam stochastic optimization

One of the well-known optimizations that are widely employed for deep learning is stochastic gradient descent (SGD) [28]. The momentum optimization [29] was introduced to help accelerate SGD by reducing its oscillations and leading it to the relevant direction, at the expense of defining an additional hyper-parameter. For this aim, we have used Adam algorithm or adaptive moments [30]. Indeed we found Adam optimization to be relatively robust for the choice of hyper parameters in our implementation. This optimization adjusts the learning rate using the first and second moments. We have set the learning rate to 0.0001 and adopted Adam’s default parameters for the first and second moments.

### 4) Image augmentation

To compensate for the lack of the number of training set of only 124 images, we need to augment the dataset. Choosing a large number of sub-images from a single image will generate overlapped information. Hence we artificially augment the training set. This would mitigate the overfitting issue and could lead to well-trained weights. For this aim, we have rotated each image from 0 to 360 degrees in 10 steps. After each rotation we select 70 sub-images. Each selected sub-image is flipped horizontally with a probability of 0.5, and then it is flipped again vertically with the same probability.

### 5) Dropout

Deep networks with many parameters are powerful tools in machine learning. However, the great number of parameters would result in overfitting issue and the failure in the network’s training procedure. To overcome this issue, we employ dropout regularization technique [31]. Using this technique, in a given layer with probability $\rho$ a subset of neurons will be dropped out, as inactive neurons. In other words, the inactive neurons would have no contribution in feedforward and back propagation processes. In such a way, since active neurons cannot rely on the dropped-out neurons they are forced to learn more robust features independently.

In addition, the dropout technique creates different thinned networks while at the test time the ensemble of all these thinned networks would produce a more efficient detection. Therefore, the network will be well-trained even when it has limited data. In this paper, as shown in Fig. 4, we use dropout with $\rho = 0.5$ after the concat layer.
V. EXPERIMENTAL RESULT

In this section, we have described the implementation details such as the used dataset and the network settings. We also present both quantitative and qualitative performance of the proposed method.

A. Implementation details

Our DFCN was tested on images from Dermquest, an online publically available dataset, accompanied by the segmentation ground truth [32]. This dataset includes 126 non-dermoscopic images consisting of 66 melanoma and 60 non-melanoma cases. The images were randomly split into four distinct folds. Images were tested based on leave-one-out cross-validation technique, through which one fold is selected for testing, two folds for training and one fold for validation at a time. In other words, 50 percent of this data-set is used for training, 25 percent for validation and the rest for testing. A set of 700 sub-images with the size of 155×155 is generated from each image. These sub-images are produced by a random vertical and horizontal flipping and rotating. Therefore, our DFCN training was performed on 44100 (i.e. 126 × 700 × 1/2) images. The network weights were initialized by Xavier method [33] while the biases in all layers were set to 0. The early stopping technique is also applied to avoid over-fitting in every iteration. The proposed method is implemented in python and Caffe [34], on a computer with an Intel Core i7-4790K processor, 32 GB of RAM, and an NVIDIA GeForce GTX Titan X GPU card.

B. Quantitative Evaluation

The DFCN outputs a probability map, in which the probability of belonging to lesion or background is determined for each pixel. By setting the threshold to 0.5, this map is converted to a binary mask. We compare the produced mask with the mask generated by expert clinician using objective measurement metrics defined in equations 8 to 12.

\[
\text{accuracy} = \frac{TP + TN}{FN + TN + TP + FP}
\]

(8)

\[
\text{sensitivity} = \frac{TP}{TP + FN}
\]

(9)

\[
\text{specificity} = \frac{TN}{TN + FP}
\]

(10)

\[
\text{dice\_score} = \frac{2TP}{2TP + FN + FP}
\]

(11)

\[
\text{border\_error} = \frac{FP + FN}{2TP + FN + FP}
\]

(12)

In these equations, TP (true positive) and the TN (true negative) are respectively the percentages of lesion and background pixels that are correctly identified. In addition, FP (false positive) and FN (false negative) are respectively the percentages of lesion and background pixels that are incorrectly identified. Figure 5 is an illustrative example which shows each of the four mentioned truly and falsely identified regions.

Using the criteria, explained in equations 8 to 12, we compare our segmentation results with those of [10], [12-16], [24], and [25]. These algorithms are tested on the Dermquest database. L-SRM [10], Otsu-R [12], Otsu-RGB [13], Otsu-PCA [14] and TDLS [15] are non-deep learning methods. Method proposed by Jafari [16] is a sliding-window based deep network. Algorithm of FCN-8s [24] has a better performance among the FCN family methods. U-net [25] is used for segmentation of medical images. Both FCN-8s and U-net are based on fully convolutional structures. Comparison results are presented in Table II.

![Fig. 5. The ground truth and the produced segmentation mask. Truly and falsely identified regions are identified.](image)

### Table II. Quantitative comparison of lesion segmentation methods applied to the Dermquest database images. Best results are bolded.

| Segmentation algorithms | Dice Score (%) | Accuracy (%) | Sensitivity (%) | Specificity (%) | Border error (%) |
|------------------------|---------------|--------------|----------------|----------------|-----------------|
| L-SRM [10]             | -             | 92.3         | 89.4           | 92.7           | -               |
| Otsu-R [12]            | -             | 84.9         | 87.3           | 85.4           | -               |
| Otsu-RGB [13]          | -             | 80.2         | 93.6           | 80.3           | -               |
| Otsu-PCA [14]          | -             | 98.1         | 79.6           | 99.6           | -               |
| TDLS [15]              | 82.8          | 98.3         | 91.2           | 99.0           | 39.7            |
| Jafari [16]            | 83.1          | 98.7         | 95.2           | 99.0           | 23.0            |
| FCN-8s [24]            | 89.7          | 98.9         | 90.0           | 99.5           | 19.0            |
| U-net [25]             | 88.7          | 98.7         | 91.5           | 99.5           | 22.4            |
| DFCN                   | **91.6**      | **98.9**     | **92.4**       | **99.6**       | **16.5**        |

As shown in Table II, our network yields 91.6% dice score that outperforms all mentioned methods. In addition, our proposed network has better accuracy, specificity and lower border error.

C. Qualitative Evaluation

In Fig 6 visual quality of our proposed method is compared with 4 other methods of Jafari [16], FCN-8s [24], U-net [25] and TDLS [15].
Fig. 6. (a) Four challenging images with ground truths shown by blue borders, (b) probability maps and segmentation results of FCN-8s [24], (c) probability maps and segmentation results of U-net [25], (d) probability maps and segmentation results of Jafari [16], (e) segmentation results of the non-deep learning method of TDLS [15], (f) probability maps and segmentation results of the proposed DFCN.
At the first row of Fig. 6 four challenging images and their ground truths are shown. Segmentation ground truths are indicated as blue border lines.

It can be seen that our method segments lesion regions more precisely in comparison to other methods. This is due to the fact that our DFCN generates precise probability maps, whereas other methods suffer from probability maps with artifacts. For example, in Fig. 6(b-1), (c-2), and (d-4) these artifacts are present.

In addition, the segmentation result of FCN, U-net and TDLS in Fig. 6(b-2), (c-3), and (e-3) are not accurate on boundaries of lesions and they misclassified the pixels that are near the borders. In fact our method generated proper probability map on lesion borders that leads to accurate boundaries.

VI. CONCLUSION

In this paper, we proposed a new class of fully convolutional networks called DFCN. We proposed dense pooling layers which preserve their features rather than losing them by greater-than-one strides. This preservation of features leads to dense resolution feature maps. Therefore our DFCN eliminates the need for a decoder phase in order to reconstruct the missing features. Meanwhile, our network is as fast as FCNs methods.

The proposed DFCN produced a dice score of 91.6% on Dermoquest database. We experimentally showed that the proposed network can outperform state-of-the-art methods in skin lesion segmentation of non-dermoscopic images. Our comparisons included well-known deep-based methods such as FCNs and U-net.

REFERENCES

[1] A. Esteva et al., “Dermatologist-level classification of skin cancer with deep neural networks,” Nature, vol. 542, no. 7639, pp. 115-118, 2017.
[2] American Cancer Society, Cancer Facts & Figures 2017, Atlanta: American Cancer Society, 2017.
[3] About Melanoma, [Online]. Available: http://polkadotmama.org/about-melanoma/. Accessed on: 14 October, 2017.
[4] Canadian Cancer Society. What is melanoma skin cancer? [Online]. http://www.cancer.ca/en/cancer-information/cancer-type/skin-melanoma/melanoma/?region=on. Accessed on: 14 October, 2017.
[5] A. F. Jerant, J. T. Johnson, C. D. Sheridan, and T. J. Caffrey, “Early detection and treatment of skin cancer.” Am. Fam. Physician., vol. 62, no. 2, pp. 357-68, 375-6, 381-2, 2000.
[6] K. Korotkov and R. Garcia, “Computerized analysis of pigmented skin lesions: a review,” Artif. Intell. Med., vol. 56, no. 2, pp. 69-90, 2012.
[7] I. Maglogiannis and C. N. Doukas, “Overview of advanced computer vision systems for skin lesions characterization,” IEEE Trans. Inf. Technol. Biomed., vol. 13, no. 5, pp. 721-733, 2009.
[8] H. C. Engasser and E. M. Warshaw, “Dermatoscopy use by US dermatologists: a cross-sectional survey,” J. Am. Acad. Dermatol., vol. 63, no. 3, pp. 412-419, 2010.
[9] M. H. Jafari et al., “Skin lesion segmentation in clinical images using deep learning,” in Proc. ICPR, 2016, pp. 337-342.
[10] M. E. Celebi et al., “Border detection in dermoscopy images using statistical region merging,” Skin. Res. Technol., vol. 14, no. 3, pp. 347-353, 2008.
[11] M. E. Celebi, H. Iyatomi, G. Schuefer, and W. V Stoecker, “Lesion border detection in dermoscopy images,” Comput. Med. Imaging Graph., vol. 33, no. 2, pp. 148-153, 2009.
[12] P. G. Cavalcanti, Y. Yari, and J. Scharcanski, “Pigmented skin lesion segmentation on macroscopic images,” in Proc. ICNZ, 2010.
[13] P. G. Cavalcanti, J. Scharcanski, and C. B. O. Lopes, “Shading attenuation in human skin color images,” Adv. Vis. Comput., vol. 6453, pp. 190-198, 2010.
[14] P. G. Cavalcanti and J. Scharcanski, “Automated prescreening of pigmented skin lesions using standard cameras,” Comput. Med. Imaging Graph., vol. 35, no. 6, pp. 481-91, 2011.
[15] J. Glaister, A. Wong, and D. A. Claudi, “Segmentation of skin lesions from digital images using joint statistical texture distinctiveness,” IEEE Trans. Biomed. Eng., vol. 61, no. 4, pp. 1220-1230, 2014.
[16] M. H. Jafari, E. Nasr-Esfahani, N. Karimi, S. M. R. Soroushmehr, S. Samavi, and K. Najarian, “Extraction of skin lesions from non-dermoscopic images for surgical excision of melanoma,” Int. J. Comput. Assist. Radiol. Surg., vol. 12, no. 6, pp. 1021-1030, 2017.
[17] J. Schmidthuber, “Deep learning in neural networks: an overview,” Neural Netw., vol. 61, pp. 85-117, 2015.
[18] G. Litjens et al., “A survey on deep learning in medical image analysis,” Med. Image Anal., vol. 42, pp. 60-88, 2017.
[19] T. Kooi et al., “Large scale deep learning for computer aided detection of mammographic lesions,” Med. Image Anal., vol. 35, pp. 303-312, 2017.
[20] M. Ghafoorian et al., “Deep multi-scale location-aware 3D convolutional neural networks for automated detection of lacunes of presumed vascular origin,” NeuroImage Clin., vol. 14, pp. 391-399, 2017.
[21] M. J. J. P. van Grinsven, B. van Ginneken, C. B. Hoyng, T. Theelen, and C. I. Sanchez, “Fast convolutional neural network training using selective data sampling: application to hemorrhage detection in color fundus images,” IEEE Trans. Med. Imaging, vol. 35, no. 5, pp. 1273-1284, 2016.
[22] W. Yang et al., “Cascade of multi-scale convolutional neural networks for bone suppression of chest radiographs in gradient domain,” Med. Image Anal., vol. 35, pp. 421-433, 2017.
[23] R. Zhao, W. Ouyang, H. Li, and X. Wang, “Saliency detection by multi-context deep learning,” in Proc. CVPR, 2015, pp.1265-1274.
[24] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in Proc. CVPR, 2015, pp. 3431-3440.
[25] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: convolutional networks for biomedical image segmentation,” in Proc. MICCAI, pp. 234-241, 2015.
[26] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A deep convolutional encoder-decoder architecture for image segmentation,” arXiv preprint arXiv:1511.00561, 2015.
[27] H. Noh, S. Hong, and B. Han, “Learning deconvolution network for semantic segmentation,” in Proc. ICCV, pp. 1520-1528, 2015.
[28] L. Bottou, “Stochastic gradient descent tricks,” Neural Networks: Tricks of the Trade, vol. 1, no. 1, pp. 1-14, 2012.
[29] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, “On the importance of initialization and momentum in deep learning,” in Proc. JMLR W&CP, 2013, pp. 1139-1147.
[30] D. P. Kingma and J. L. Ba, “Adam: a method for stochastic optimization,” in Proc. ICLR, 2015, pp. 1-15.
[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” J. Mach. Learn. Res., vol. 15, pp. 1929-1958, 2014.
[32] Vision and Image Processing Lab, [Online]. Available: http://vip.uwaterloo.ca/demos/skin-cancer-detection. Accessed on: 29 October, 2017.
[33] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in Proc. AISTATS, pp. 249-256, 2010.
[34] Y. Jia et al., “Caffe: convolutional architecture for fast feature embedding,” in Proc. ACM-MM, 2014, pp. 675-678.