Binarization of Color Character Strings in Scene Images Using Deep Neural Network

Bian, Wenjiao

(出版者 / Publisher)
法政大学大学院情報科学研究科
(雑誌名 / Journal or Publication Title)
法政大学大学院紀要. 情報科学研究科編
(巻 / Volume)
13
(開始ページ / Start Page)
1
(終了ページ / End Page)
6
(発行年 / Year)
2017-03-31
(URL)
https://doi.org/10.15002/00021511

Binarization of Color Character Strings in Scene Images Using Deep Neural Network

Wenjiao Bian
Graduate School of Computer and Information Sciences
Hosei University
3-7-2 Kajino-cho, Koganei-shi, Tokyo 184-8584, Japan
E-mail: wenjiao.bian.44@stu.hosei.ac.jp

***Abstract*—This paper addresses the problem of binarizing multicolored character strings in scene images with complex backgrounds and heavy image degradations. The proposed method consists of three steps. The first step is combinatorial generation of binarized images via every dichotomization of *K* clusters obtained by K-means clustering of constituent pixels of an input image in the HSI color space. The second step is classification of each binarized image using deep neural network into two categories: character string and non-character string. The final step is selection of a single binarized image with the highest degree of character string as an optimal binarization result. Experimental results using ICDAR 2003 robust word recognition dataset show that the proposed method achieves a correct binarization rate of 87.4% that is highly competitive with the state of the art of binarization of scene character strings.**

Keywords—binarization of color character strings; K-means clustering; deep neural network; convolutional neural network

I. INTRODUCTION

In recent years, recognition of scene image text is becoming a crucial and hot topic due to the widespread use of smart phones and social network, the huge demands for Internet image search, and the emerging need for understanding of dynamic videos. In the process of color character stings recognition in scene images, there are mainly three key steps: localization of character strings, binarization of color character strings, and distortion-tolerant character recognition. In particular, the performance of correct binarization has a great influence on the accuracy of recognition.

Traditional binarization methods proposed appropriate selection of binarization threshold: global threshold selection [1] [2], local threshold selection [3] [4], and combination of local and global thresholds [5]. On the other hand, without using threshold Wakahara et al. [6] proposed to binarize multicolored scene character strings by combinatorial generation of binarized images via K-means clustering and selection of optimally binarized image using support vector machines. Although we have already many promising methods, the topic of binarization of color character images still poses a considerable challenge due to heavy image degradations such as blur and distortion, a wide variety of complex backgrounds and lightening conditions, and existence of various kinds of multicolored characters.

Deep neural network has achieved an enormous success in many fields of computer vision and pattern recognition in recently years. However, an application of deep neural network to the image binarization process does not exist to the best of our knowledge. Considering the power of deep neural network, we propose a new method to binarize color character strings in scene images in this paper.

Our proposed method consists of three steps. The first step is combinatorial generation of binarized images via every dichotomization of *K* clusters obtained by K-means clustering applied to constituent pixels of an input image of character string in the HSI color space. This step follows the same idea proposed by Wakahara et al. [6]. The second step is classification of each binarized image using deep neural network into two categories: character string and non-character string. We proposed two kinds of deep neural network. One is deep neural network composed of affine layers using feature vectors as input. We extract features of black pixel ratio and crossing count from each binarized image. The other is deep neural network composed of convolutional layers using grayscale values as input. The final step is selection of a single binarized image with the highest degree of character string as an optimal binarization result of the input color image.

Fig.1 shows the total flow of our proposed method.

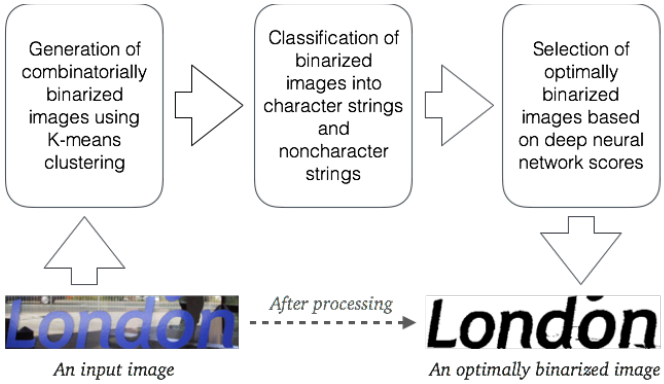


Fig. 1. Total flow of our proposed method.

In our experiments, we used the well-known ICDAR 2003 robust word recognition dataset [7]. Experimental results show that the proposed method achieves a correct binarization rate of 87.4% that is highly competitive with the state of the art of binarization of scene character strings.

Concretely, our contributions are as follows. We proposed a new method which applies deep neural network in the process

of scene character string binarization. This kind of application does not exist to the best of our knowledge. And our proposed method is highly competitive with the state of the art of binarization method.

This paper is organized as follows. In Section II, we describe the dataset used in our experiments. In Section III, we explain the details of how K-means clustering can be used to generate combinatorially binarized images. Then, in Section IV, we show how to classify binarized images into two categories: character strings and non-character strings using deep neural network. In Section V, we propose to select a single binarized image as an optimal binarization result using deep neural network scores representing the degree of character string. What's more, in Section VI, we show experimental results of our proposed method and compare it against the state of the art of binarization of scene character strings. Finally, Section VII concludes this paper by pointing out future work.

II. ICDAR 2003 Robust Word Recognition Dataset

There are several well-known datasets used in ICDAR 2003 robust reading competitions [8]. In our experiments, we choose the robust word recognition dataset [7] containing a set of JPEG images of single words in natural scenes. The numbers of images in "TrialTrain" subset and "TrialTest" subset are 1156 and 340, respectively.

Fig. 2 shows some examples used in our experiments. We can clearly see that the multicolored character strings in scene images are subject to heavy image degradations and complex backgrounds.

Fig. 2. Examples of color character strings used in our experiments.

III. Generation of Combinatorially Binarized Images Using K-Means Clustering

In this step, we apply K-means clustering to constituent pixels of a given color image of character string in the HSI color space, and generate combinatorially binarized images via every dichotomization of a total of K clusters [6].

First, we convert a RGB color space image into HSI color space and scale each value of H, S, and I to range from 0 to 255. Preliminary experiments showed that the conversion from the RGB color space to the HSI color space was useful for contrasting characters against backgrounds.

Second, we apply K-means clustering to generate K clusters, where the number of clusters, K, should be determined in advance. The value of K should be of an appropriate size. A too small value of K will may fail to generate a correctly binarized image. On the other hand, a too large value of K will result in generation of superabundant binarized images and make it difficult to train the deep neural network that classifies binarized images into character string and non-character string.

Finally, we divide K clusters into two groups, and set values of pixels in the one group at 0 (black) and set values of pixels in the other group at 255 (white). As a result, we get a binarized image, where black pixels are assumed to represent a character string while white pixels are assumed to represent background. By considering every possible dichotomization of K clusters we can generate combinatorially binarized images the total number of which, N_{binary}, is given by

$$N_{binary} = 2^K - 2. \quad (1)$$

Fig.3 shows one example of generation of combinatorially binarized images using K-means clustering and every possible dichotomization of K clusters.

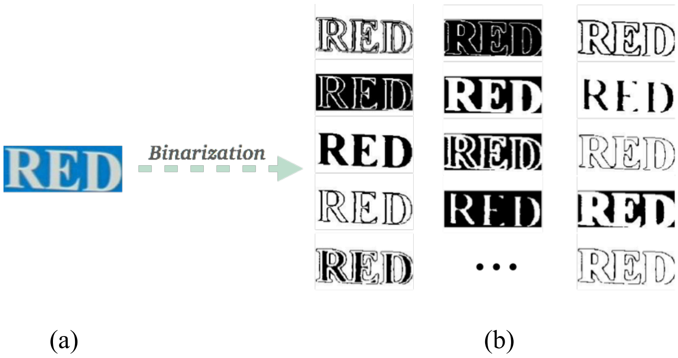


Fig. 3. One example of generation of combinatorially binarized images by K-means clustering. (a) Original color image. (b) Binarized images generated by every possible dichotomization of K clusters.

IV. Classification of Binarized Images into Character Strings and Non-Character Strings

A. Feature extraction from binarized images

The character string images in the ICDAR 2003 robust word recognition dataset vary greatly in width and height. However, the number of inputs in the first layer of deep neural network should be definite. Therefore, we need a kind of image size normalization as preprocessing. From this viewpoint, we investigated the ratios of width to height using all images in the "TrialTrain" subset, and calculated the average width/height ratio defined by

$$R_{wh} = \frac{1}{N}\sum_{i=1}^{N}\frac{w_i}{h_i}, \quad (2)$$

where N is the total number of images in the dataset.

As a result, we found that the average width/height ratio of character string images in the dataset was around 3.4. Hence, we decided that the normalized image should have the width of 170 pixels and the height of 50 pixels. Actually, we applied the above-mentioned size normalization to all images in both "TrialTrain" and "TrialTest" subsets by means of bilinear interpolation.

Fig. 4 shows an example of size normalization of binarized images.

(a) (b)

Fig. 4. Size normalization of binarized images. (a) Original image with 232×831 pixels. (b) Normalized image with with 50×170 pixels.

We build two kinds of deep neural network for experiments in this paper: one is composed of affine layers using black pixel ratio and crossing count feature vectors as input; another is composed of convolutional layers using grayscale values as input. The black pixel ratio is the ratio of black pixels in each row or column. The crossing count is a total count of transitions from black to white pixels or white to black pixels in each row or column.

Fig. 5 and Fig. 6 show how to extract features of black pixel ratio and crossing count, respectively, from a binarized image.

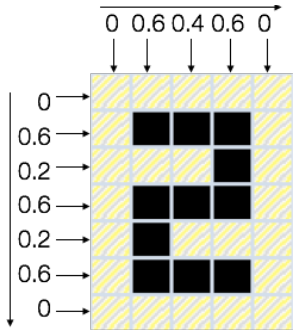


Fig. 5. Extraction of black pixel ratio feature.

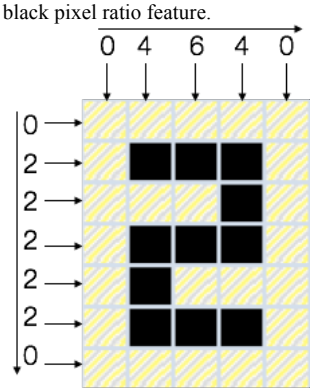


Fig. 6. Extraction of crossing count feature.

B. *Deep neural network composed of affine layers using feature vectors as input*

We build a four-hidden-layer deep neural network (DNN) for using black pixel ratio (BPR) feature, crossing count (CC) feature as input. All units in hidden layers are equipped with ReLU as a nonlinear activation function [9]. We adopt a batch normalization and a dropout regularization for each fully-connected affine layer. The final output layer is a softmax layer with two units: one unit is assigned to non-character string class (negative), and the other unit is assigned to character string class (positive).

Fig. 7 illustrates the structure of deep neural network composed of multiple affine layers.

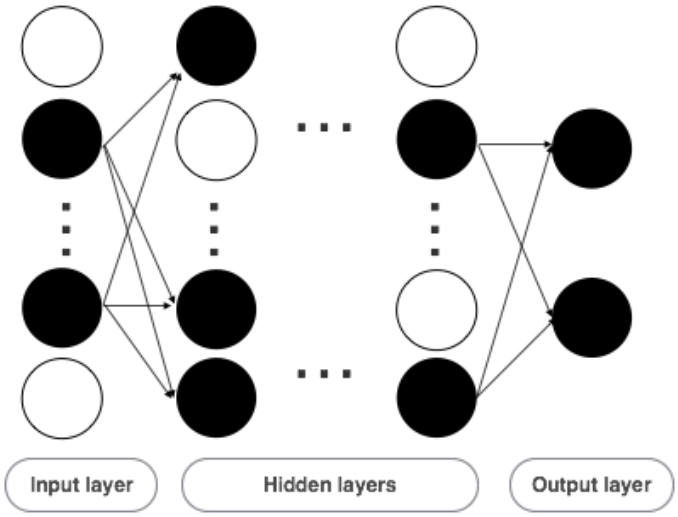
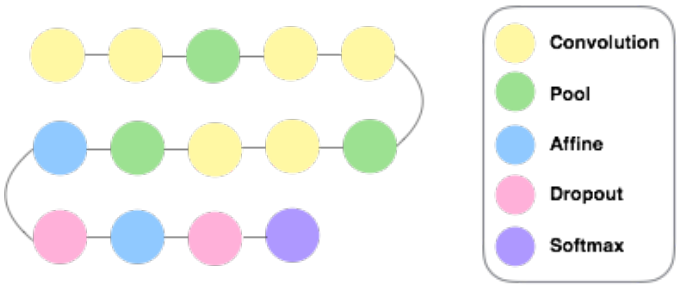


Fig. 7. Structure of deep neural network composed of multiple affine layers. White circles denote dropping out units while black circles denote active ones.

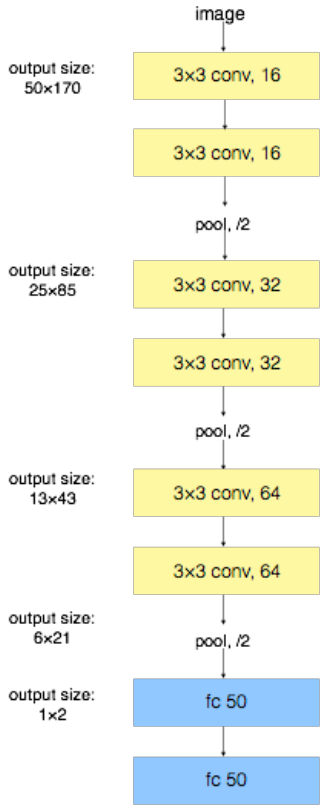
C. *Deep neural network composed of convolutional layers using grayscale values as input*

First, a 50×170 grayscale image is fed into an input layer of deep neural network. Then, we use a stack of convolutional layers, where the size of each convolutional filter is set at 3×3 [10] fixed filter size. Also, we adopt a convolution stride of 1, together with a zero padding of size 1 for 3×3 convolutional filters. A pooling layer performs max-pooling using a 2×2 pixel window with a stride of 2. In hidden layers ReLU layers realize nonlinear activation. Furthermore, we use a dropout regularization for each fully-connected affine layer. The final output layer is the softmax layer. We build deep neural network composed of six convolutional layers, three pooling layers, two affine layers, two dropout layers, and one softmax layer. The final output layer is a softmax layer with two units.

Fig. 8 shows the structure of our CNN-based deep neural network. Functions of constituent layers are distinguished by their colors.



(a)



(b)

Fig. 8. Structure of deep neural network composed of multiple convolutional layers. (a) The sequence of constituent layers. (b) The detailed specification of our proposed CNN.

D. *Training of deep neural networks*

Regarding BPR or CC based DNN, the size of inputs is 220 (= 50 +170) and the number of hidden units in each hidden layer is 75. When we combine BPR and CC features the size of inputs amounts to 440 (= 220+220) and the number of hidden units is 150. A mini-batch gradient descent with Adam optimizer [11] is used. In order to find the suitable parameters for deep neural network, we choose a set of varied parameters for testing. After several tests, we choose a group of parameters that achieved the best performance on cross validation subset. The mini-batch size is set at 100; dropout ratio is set at 0.25. The number of learning epochs is set at 500.

On the other hand, regarding grayscale value based CNN the size of inputs is 8500 (= 50×170). The mini-batch size is set at 100; dropout ratio is set at 0.5. The number of learning epochs is set at 20.

V. Selection of Optimally Binarized Images Based on Deep Neural Network Scores

As described in Section III, we generate a total of $(2^K - 2)$ binarized images for one original color image. Actually, we set the number of clusters, K, at 5. As a result, we have 30 binarized images. On the other hand, as described in Section IV, our proposed deep neural network outputs a pair of scores, s_1, s_2 ; s_1 represents the degree of "non-character string" likeness while s_2 represents the degree of "character string" likeness. When s_1 is larger than s_2, we classify the binarized image as non-character string, and vice versa. Therefore, we propose to select the binarized image, I^q, with the highest score of s_2 among a total 30 binarized images, $\{I^k\}_{k=1}^{30}$, as the optimally binarized image according to the following formula.

$$q = \underset{k}{\operatorname{argmax}}\, s_2^k \tag{3}$$

VI. Experimental Results

A. *Training and testing data with correct classification tags*

As described in Section I, we used ICDAR 2003 robust word recognition dataset. We use 1156 images in "TrialTrain" subset for training. On the other hand, for testing we use 340 images in "TrialTest" subset. The value of K in K-means clustering was set at 5. Therefore, we get 30 binarized images for each original color image.

At first, we manually gave each of all binarized images its corresponding correct classification tag, 1 or 0; the tag "1" represents "positive" (character string), and the tag "0" represents "negative" (non-character string). Actually, in the training set we had 1962 "positives" and 32718 "negatives" among 1156×30 binarized images. In the testing set we had 340 "positives" and 9860 "negatives" among 340×30 binarized images.

Then, we used the training set to train our proposed deep neural networks. It is to be noted that we used "positives" repetitively to achieve a balance between the numbers of "positives" and "negatives." As a result, we provided 31382 "positives" and 32718 "negatives" for training. We trained four kinds of deep neural networks: BPR based neural network, CC based one, combined BPR+CC based one, and CNN.

Table I shows comparison of time required for DNN training. We used NVIDIA GeForce® GTX 760 GPU in experiments.

From Table I, we can say that CNN needs much longer time than the other three. Hence, if we just want a fast training method, we can choose a combination of BPR and CC based DNN instead of time-consuming CNN.

TABLE I. Comparison of DNN Training Time

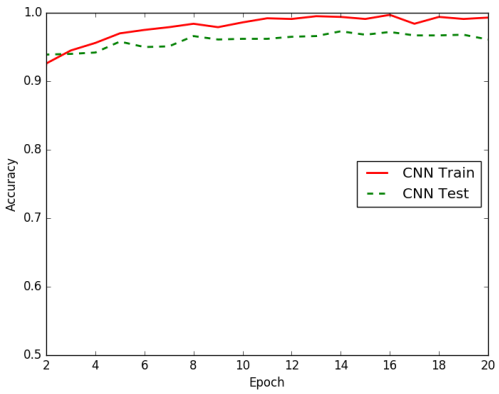
Method	Training time (hrs.)
BPR	0.5
CC	0.5
BPR+CC	0.5
CNN	40

B. *Accuracy of classifying combinatorially binarized images into character strings and non-character strings*

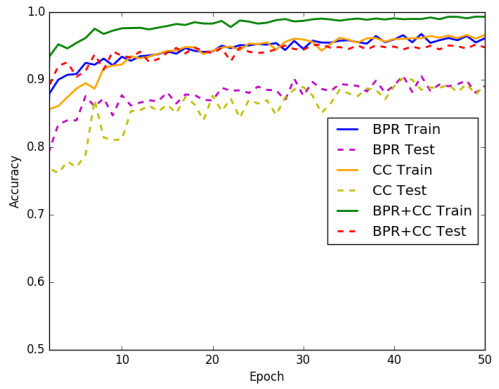
We applied the above-mentioned four kinds of deep neural networks to classification of binarized images generated from both the training set and the testing set into character string and non-character string.

Fig. 9 shows relations between classification accuracy and the number of epochs used for training of deep neural networks.

From Fig. 9 (a), it is clear that classification accuracy of CNN improves steadily as the epoch number increases and reaches to a kind of saturation point for testing after ten epochs. Also, from Fig. 9 (b), we can see that the classification accuracy of BPR and CC fluctuates violently especially for testing. This fact means that these two kinds of features may not be so effective in discriminating between character string and non-character string. However, a combined BPR+CC achieved a fairly stable and moderate classification accuracy as compared to BRR and CC.



(a)



(b)

Fig. 9. Relations between classification accuracy and the number of epochs used for training. (a) CNN. (b) BPR, CC, and BPR+CC.

Table II summarizes the classification accuracy for testing obtained by the four kinds of deep neural networks.

From Table II, it is clear that CNN achieved the highest accuracy of 96.5%. Also, we find that although feature-based deep neural networks are decidedly inferior to CNN, a combined BPR+CC is narrowly comparable to CNN.

TABLE II. CLASSIFICATION ACCURACY

Method	Classification accuracy (%)
BPR	91.8
CC	91.3
BPR+CC	95.6
CNN	**96.5**

C. *Accuracy of selecting optimally binarized images*

Following the procedure described in Section V we evaluated the accuracy of selecting optimally binarized images for testing. The total number of testing color images were 340.

Table III shows the selection accuracy obtained by the four kinds of deep neural networks: BPR, CC, BPR+CC, and CNN.

From Table III, it is found that CNN achieved the highest rate of 87.4% that is far superior to those obtained by feature-based deep neural networks. Moreover, the selection accuracy of CNN definitely outperformed the competing method [6] that utilized SVM for selecting optimally binarized images.

TABLE III. SELECTION ACCURACY

Method	Selection accuracy (%)
BPR	64.1
CC	66.2
BPR+CC	80.6
CNN	**87.4**
Wakahara et al. [6]	80.4

Fig. 10 shows some examples of correctly binarized images by the proposed method based on CNN.

From Fig. 10, we can say that the proposed binarization method is robust against a variety of lighting conditions, complex backgrounds, and heavy image degradation.

We carefully examined and analyzed mistakenly binarized images by the proposed method and found that two major causes of incorrect binarization by the proposed method were in the following.

1st cause: Generation of combinatorially binarized images by K-means clustering can not generate an optimal binarized image.

2nd cause: Our proposed deep neural networks fail to select optimally binarized images.

Fig. 11 and Fig. 12 show examples of mistakenly binarized images due to the above-mentioned two causes, respectively.

(a) (b)

Fig. 10. Examples of correctly binarized images by the proposed method. (a) Original color images. (b) Binarized images.

(a) (b)

Fig. 11. Examples of mistakenly binarized images due to the first cause. (a) Original color images. (b) Binarized images.

(a) (b) (c)

Fig. 12. Examples of mistakenly binarized images due to the second cause. (a) Original color images. (b) Selected binarized images. (c) Optimally binarized images.

Considering the above-mentioned two major causes of incorrect binarization we enumerate future topics as follows.

(1) Adaptive selection of an appropriate number, K, of clusters according to a variety of colors in an input image to generate a necessary and sufficient number of binarized images.

(2) Use of more advanced deep neural networks [12] to discriminate between character string and non-character string with a high accuracy.

VII. Conclusion

We have proposed a new, deep neural network based technique of binarizing multicolored character strings in scene images. Namely, by using K-means clustering, we generate combinatorially binarized images, and classify each binarized image into two categories: character string and non-character string via deep neural network. In particular, we tried two kinds of deep neural networks: feature-based multi-layer network and convolutional network. These key ideas clearly distinguish our proposed method from those conventional techniques of binarization heavily relying on threshold selection. Experimental results using ICDAR 2003 robust word recognition dataset show that our proposed method achieves a high accuracy of 87.4% on binarization of multicolored character strings in scene images with complex backgrounds and heavy image degradations. As future work, we can easily extend our model to some more advanced deep neural network structures to achieve a higher accuracy.

References

[1] N. Otsu, "A threshold selection method from gray level histograms," IEEE Trans. Systems, Man and Cybernetics, vol. 9, pp. 62–66, 1979.

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

[3] J. Sauvola and M. Pietikinen, "Adaptive document image binarization," Pattern Recognition, vol. 33, pp. 225–236, 2000.

[4] W. Niblack, An introduction to digital image processing. Prentice Hall, 1986.

[5] S. Milyaev, O. Barinova, T. Novikova, P. Kohli, and V. Lempitsky, "Image binarization for end-to-end text understanding in natural images," Proc. of the 12th Int. Conf. on Document Analysis and Recognition (ICDAR2013), pp. 128-132, Washington, Aug. 2013.

[6] T. Wakahara and K. Kita, "Binarization of color character strings in scene images using k-means clustering and support vector machines," Proc. of the 11th Int. Conf. on Document Analysis and Recognition, pp. 274-278, Beijing, Sept. 2011.

[7] The ICDAR 2003 Robust Reading Datasets. http://algoval.essex.ac.uk/icdar/Datasets.html.

[8] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong and R. Young, "ICDAR 2003 Robust Reading Competitions," Proc. of the 7th Int. Conf. Document Analysis and Recognition (ICDAR 2003), pp. 1-6, Edinburgh, Aug. 2003.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Proc. of the 25th Int. Conf. on Neural Information Processing Systems (NIPS'12), pp. 1097–1105, Lake Tahoe, Dec. 2012.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", arXiv:1409.1556 [cs.CV], Sept. 2014.

[11] D. P. Kingma, J. Ba, "Adam: a method for stochastic optimization", arXiv:1412.6980 [cs.LG], Dec. 2015

[12] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition", arXiv:1512.03385 [cs.CV], Dec. 2015.