Binarization strategy using multiple convolutional autoencoder network for old Sundanese manuscript images

Paulus, E.; Burie, J-C.; Verbeek, F.J.; Barney Smith, E.H.; Pal, U.

Binarization Strategy Using Multiple Convolutional Autoencoder Network for Old Sundanese Manuscript Images

Erick Paulus[1,3(✉)], Jean-Christophe Burie[2], and Fons J. Verbeek[1]

[1] Leiden Institute of Advanced Computer Science, Leiden University, Leiden, The Netherlands
{e.paulus,f.j.verbeek}@liacs.leidenuniv.nl
[2] Laboratoire Informatique Image Interaction (L3i) University of La Rochelle, Avenue Michel Crépeau, 17042 La Rochelle Cedex 1, France
jcburie@univ-lr.fr
[3] Computer Science Department, Universitas Padjadajaran, Bandung, Indonesia
erick.paulus@unpad.ac.id

Abstract. The binarization step for old documents is still a challenging task even though many hand-engineered and deep learning algorithms have been offered. In this research work, we address foreground and background segmentation using a convolutional autoencoder network with 3 supporting components. The assessment of several hyper-parameters including the window size, the number of convolution layers, the kernel size, the number of filters as well as the number of encoder-decoder layers on the network is conducted. In addition, the skip connections approach is considered in the decoding procedure. Moreover, we evaluated the summation and concatenation function before the up-sampling process to reuse the previous low-level feature maps and to enrich the decoded representation. Based on several experiments, we determined that kernel size, the number of filters, and the number of encoder-decoder blocks have a little impact in term of binarization performance. While the window size and multiple convolutional layers are more impactful than other hyper-parameters. However, they require more storage and may increase computation costs. Moreover, a careful embedding of batch normalization and dropout layers also provides a contribution to handle overfitting in the deep learning model. Overall, the multiple convolutional autoencoder network with skip connection successfully enhances the binarization accuracy on old Sundanese palm leaf manuscripts compared to preceding state of the art methods.

Keywords: Binarization · Autoencoder · Palm leaf manuscript · Deep learning

1 Introduction

In document image analysis (DIA), the binarization task is responsible for determining which pixels belong to the character and which to the background. As a

E. H. Barney Smith and U. Pal (Eds.): ICDAR 2021 Workshops, LNCS 12917, pp. 142–157, 2021.
https://doi.org/10.1007/978-3-030-86159-9_10

result, binarization plays a key role on the performance of the OCR. Nonetheless, there are numerous difficulties in binarizing old document images, including fading text, age color change, crack, and bleed-through noises [1]. Furthermore, the binarization problems are exacerbated by a variety of tools and acquisition methods. Owing to inadequate lighting and camera blitz, the image quality may suffer from blur, low contrast, and uneven illumination [2].

Many works to solve these problems have been proposed in the literature. Initially, several researchers advocated hand-engineered solutions that relied on image characteristics. The global thresholding procedure is one of the most basic binarization methods. The Otsu algorithm [3] is the most widely used thresholding technique and is commonly used as a benchmarking tool. When dealing with degraded images, a single threshold seldom produces acceptable binarization results. Since 1986, researchers have been pursuing local adaptive thresholding, for example, Niblack's [4] and Bernsen's [5]. It has an advantage in finding the adaptable threshold per patch or window. Some authors utilize histograms and statistics to gain a better threshold by calculating the minimum, maximum, mean, and standard deviation of intensity for each window. Another advantage of these handcrafted methods is that they are less expensive to compute because they only use the image characteristics and do not need a labelled image to be trained. They also successfully yield a good binarized image over uniform and low contrast problems. Nonetheless, local adaptive thresholding algorithms are not enough to tackle images suffering from uneven illumination. For example, the F-measure of hand-engineered binarization approaches on old Sundanese manuscripts, only reached a score of 46.79 [6].

Another challenge of traditional adaptive thresholding is that there are several parameters to be adjusted to achieve better performance. As a result, it is regarded as a disadvantage. Several methods for automatically adjusting those parameters have been proposed and evaluated. Howe improved his previous algorithm by incorporating automatic parameter tuning with pairwise Canny-based approach [7]. Later, Mesquita optimized Howe's method using a racing algorithm [8]. Kaur et al. improved Sauvola's thresholding [9] by applying stroke width transform (SWT) to automatically calculate window size.

The advent of deep learning algorithms allowing machine work with or without any supervision has opened a new gate to address more complex problems and tune automatically parameter. In 2015, Pastor et al. first introduced the CNN implementation on binarization task by classifying every pixel into background and foreground based on a sliding window [10]. In the meantime, a transformation of CNN into a Fully Convolutional Networks (FCN) topology was announced for the semantic segmentation task where every pixel is determined as one of K classes [11,12]. Later, Tensmeyer et al. defined binarization likewise a semantic segmentation with $K = 2$ and gained a better performance compared to CNN [13]. Recently, a combination of an hourglass-shaped deep neural network (DNN) and convolution network was proposed to handle binarization challenges. It is named as Convolutional Auto Encoder (CAE) network that has the main advantage to convolve image-to-image as well as to learn compressed and

distributed encoding (representation) from a training set. Furthermore, in [14,15], a composite of residual block and CAE outperformed several prior DNN especially on LRDE DBD [16], DIBCO [17], and palm leaf manuscript [18] datasets.

In this paper, we firstly aimed to assess the contribution of five parameters in CAE and three supporting components including the window size, the number of encoder-decoder layers, the number of convolution layers, the number of filters, the kernel size, the skip connector, the dropout size, and the existence of the batch normalization to achieve the better binarization performance. Second, we aimed to assess the effectiveness of the CAE network to deal with binarization challenges on old Sundanese palm leaf manuscripts. The main contribution of this work is to propose a strategy to determine the best parameters to configure CAE. We present a set of comprehensive experiments analyzing in details the influence of both hyper-parameters of CAE and supporting components. So that, the efficacy of each hyper-parameter can be exploited to build an appropriate CAE network for processing the images of our Old Sundanese palm leaf manuscripts. As far we know, to the proposed CAE network in this paper is the first architecture with several convolution layers for each encoder-decoder block in a hour-glass model.

This paper is organized as follow. The second section describes the related works in terms of CAE network, skip connection, dropout, and batch normalization. Experiment procedures and evaluation metrics are detailed in the third section. Then, the fourth section presents the results of the experiments. Finally, the conclusions and future research topics are discussed in the last section.

2 Related Work

2.1 Convolutional Autoencoder Network

Autoencoder is a type of unsupervised neural network which can learn latent representation from input data [19]. In this case, it does not need a labeled training set. Autoencoder is also well-known as a dimensionality reduction algorithm, feature detector, generative model, or unsupervised pretraining of deep learning model. Several implementations of this neural network in computer vision area are image denoising [20], augmentation data [20], anomaly detection [21,22], image restoration [23] and invertible grayscale [24]. In another case, the execution of autoencoder in a two-level segmentation task or binarization task still needs labeled image or ground truth binary image. In other words, the weights of this deep neural network architecture are gained from supervised learning.

In general, an autoencoder is made up of three parts: an encoder part, an internal hidden layer (the code part), and a decoder part. The encoder is responsible for compressing the input and producing the code or latent representation, whereas the decoder is responsible for rebuilding or unzipping the code part provided by the hidden layer. Furthermore, autoencoders use fully-connected feedforward neural networks for training, following the same procedure as backpropagation.

Several types of autoencoders have been proposed successively, including simple autoencoders, convolutional autoencoders (CAE), and variational autoencoders. The similarity of which is that the dimension of output and input are always same. Nonetheless, the different types of autoencoders can be distinguished by the manner in which the latent representation (the code part) is constructed with the input data, then used to reconstruct the output data. The simple autoencoder employs a dense network, which means that each output is linearly connected to each input. The convolution autoencoder can use convolution operators to generate a low-level representation feature map, which can then be used to analyze the images. So, the spatial information of the image is preserved, which can be useful during the training step. Variational autoencoders propose a learning process based on a probability distribution function by imposing additional constraints on the encoded representations using Bayesian inference [25]. The purpose of a variational autoencoder is to serve as a generative model with the ability to generate new data instances. Nonetheless, the principle is much easier to understand than those of Generative Adversarial Networks.

In this study, we focus the use of the convolutional autoencoder on the binarization task. We decompose the CAE for experimental purposes into convolution blocks, deconvolution blocks, encoder blocks, and decoder blocks. As shown in Fig. 1, the decoder block is generally mirrored to the encoder block. Each encoder block employs a convolution block with a stride of 2, in order to compress the input by dividing the dimension of the output by two. During the model training process, each decoder block uses transposed convolution with the same stride as the encoder to up sample the input and learn how to fill in missing parts. In addition, some researchers demonstrated the use of a pooling layer for down sampling in the encoder and an up-sampling method in the decoder, as explained by [22], allows to solve the problem of anomaly image detection. In the case of image restoration using the Residual Encoder-Decoder Network (RED-Net), a pooling layer is not recommended for down sampling because deconvolution in the decoder does not work well [23]. However, the use of strided convolution and pooling layers is still debatable. Because each brings its own set of benefits and drawbacks. [26] proposes another study on strided convolution layer and pooling layer.

Aside from stride, several parameters in the convolution and deconvolution blocks must be configured: the number of filters, kernel size, padding type, and activation function. RED-Net advised to configure layers with a kernel size of 3×3 and 64 filters [23]. Meanwhile, Selectional Auto-Encoder (SAE) recommended that the convolution block and deconvolution be fixed with kernel sizes of 5×5 and 64 filters [15]. In addition, both blocks have a rectified linear activation ReLU and padding "same".

In the context of the encoder-decoder block, we must consider the number of encoder and decoder blocks, as well as the number of convolution/deconvolution within them. In a typical CAE, one convolution is used for each encoder and decoder block. Indeed, in convolution/deconvolution, these parameters are closely related to the input size and stride size. For instance, if there are five

encoder blocks and the size of the squared input is 256, the size of the encoded representation after the fifth encoding with stride 2 in each convolution layer is 8×8. If the same configuration is used, but the number of encoders increases to eight, the size of the encoded representation in the final encoding is 1×1. The information in the latent representation is now meaningless. Fixing the larger patch size, on the other hand, will increase the computation cost. Furthermore, the patch should be smaller in size than the image. As a result, we need to manage them carefully.

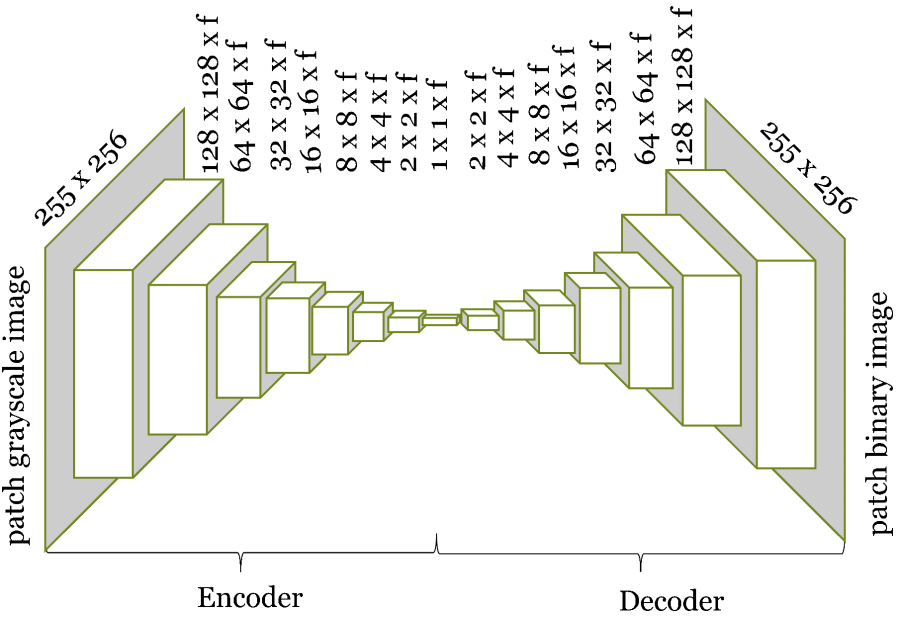

Fig. 1. The standard architecture of convolutional autoencoder.

2.2 Skip Connections Method

The skip connection is used for two reasons [23]. First, the feature banks delivered by skip connections contain significantly more information, which will be useful for obtaining an enriched representation in the next layer. Second, the skip connection in the deeper network allows training easier.

Concatenation and summation methods are the two most common skip connection variants. ResNet [27], for example, uses element-wise summation as a standard skip-connection architecture. It enables the features to be refined as it moves through the network's various layers. The number of features is preserved by using additional skip-connections in which the feature dimensions are the same before and after the skip-connection. Residual Encoder Decoder Network

(RED-Net) [23] and Selectional Auto Encoder (SAE) [15] are two other deep learning architectures that use the summation method.

Concatenated skip connections, on the other hand, increase the dimension of the representations. Nonetheless, it allows to reuse representations from previous layers. Concatenation's main advantage is that it provides more information, which is useful in deep supervision and may result in better gradient propagation. DenseNet [28] and Inception [29] are two samples of deep learning architecture which utilize the concatenation approach.

2.3 Dropout Regularization

Aside from incurring higher computation costs, training in a deeper network with large hyperparameters may result in an overfitting problem. Dropout is a regularization technique that is commonly used to overcome this problem. The main idea is to drop units at random during training to avoid excessive coadaptation [30]. The probability of dropping a unit is a float value in range 0 and 1. The drop value of 0 indicates that all units are preserved. However, the higher the dropout, the smaller the unit. Dropout can be placed in either the input or hidden layers. Dropout close to 0 is ideal for the input layer, while dropout between 0.2 and 0.5 is ideal for the hidden layer.

2.4 The Batch Normalization

To address the training speed issue, Ioffe and Szegedy developed a batch normalization (BN) mechanism that reduced internal covariate shift. BN has several advantages, including the use of higher learning rates, a low intention on initialization, and, in some cases, a reduction in the dropout layer [31]. They demonstrated some promising results with the corresponding to BN and Dropout experiments into Inception model. Many current deep learning architectures, including [32] and [33], use BN because of its ability to reduce training time while avoiding overfitting. An empirical study of BN over several deep neural networks was also presented in [34].

Other arguments with respect to BN and Dropout mechanisms are found in the literature. Garbin et al. presented a comprehensive comparison study between them on multilayer perceptron networks and convolutional neural networks (ConvNet) [35]. As practical guidelines for ConvNet, BN can improve accuracy with a short training period and should be the first consideration when optimizing a CNN. Dropout regularization, on the other hand, should be used with caution because it can reduce accuracy.

3 Dataset and Methodology

3.1 Datasets

In our experimental study, we used old Sundanese palm leaf manuscripts written by different scribers. The data set consists in 61 images split in a training set of 31

images and a testing set of 30 images. This dataset is the one used for the ICFHR 2018 competition [36]. It allows to compare our results to the methods presented in the competition. In addition, each image has its own ground truth image, which was manually created using Pixlabeler software by a group of philologists and students from the Sundanese Department at Universitas Padjadjaran in Indonesia.

3.2 Experiment Procedures

We investigate the impact of five CAE architecture parameters and three other components on binarization performance. We conduct the experiments on old Sundanese palm leaf manuscripts. Note that these experiments do not use augmented data. As a starting point, we followed the SAE configuration recommendation [15]. Instead of processing the image as a whole, image data is trained locally using a patch-wise approach. The weights of network are initialized based on Xavier weight initialization to manage the variance of activations and back-propagated gradients, resulting in faster convergence and higher accuracy. They are then optimized using stochastic gradient descent, a batch size of 10, and an adaptive learning rate of 0.001 [37]. The training step iteration is set to 200 epochs. However, if there is no decrease in training loss after 20 epochs, the early stopping strategy is executed. Several configurations are available, including a square window with a side size of 256, 64 filters and a kernel size of five for each convolution layer, five encoder blocks and five decoder blocks, a skip connection for each decoder block, BN, and a dropout of 0.

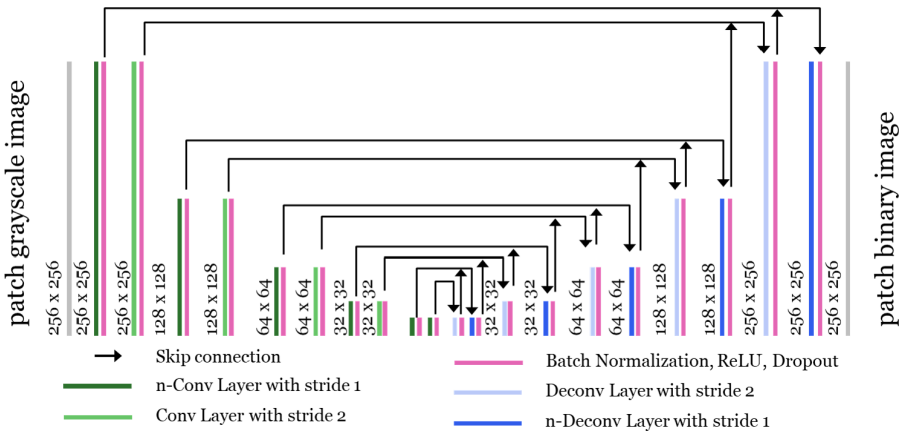

Fig. 2. The architecture of CAE network with its supporting components.

The first experiment scenario focus on CAE hyperparameters. We set five window sizes, which is successively to 64, 128, 256, 320, and 384. Then, we take the best two window sizes and change the number of filters to 16, 32, 64, 96,

and 128 while keeping the kernel size at 3×3 and 7×7. Several modifications to the number of encoder-decoder blocks is simulated based on the best two previous experiments. Afterward, we evaluated the number of strided convolution/deconvolution inside encoder-decoder block. The rectified linear activation function (ReLU) and padding "same" are applied in all convolution and deconvolution in this scenario. The position of each CAE layer and its supporting components is depicted in Fig. 2.

Secondly, a review of supporting components is carried out. We begin by varying the dropout value between 0.1 and 0.5 with and without BN. The existence of skip-connections for both summation and concatenation is then evaluated using the best two results of the previous configurations.

3.3 Evaluation Method

During the training process, F-measure (Fm) is used to evaluate the CAE architecture, because this measure is a popular metric for imbalanced classification [38]. In the context of binarization, this means that the number of background pixels is much greater than the number of foreground pixels.

4 Result and Discussion

All experiments are written in Python programming language and supported by packages such as Scikit-Learn, Numpy, Keras, and Tensorflow. In terms of hardware, we use a GeForce 2x RTX 2070 (8 GB/GPU) Graphics Processing Unit (GPU) and 48 GB RAM. All discussions are based on the binarized images predicted by the best-weighted model on the testing dataset.

4.1 Discussion on Five Hyper Parameters Experiments

In the first experiment we assess the influence of window size of input image and the numbers of filters in CAE model which apply three supporting components. As shown in Fig. 3, we evaluated five different squared window sizes ranging from 64×64 to 384×384 in four different numbers of filters (f16, f32, f64, and f96). The box size of 384 had the lowest accuracy. Because, the boxes are too wide or long in comparison to the original image which may also result in poor performance. While boxes with sides of 64, 128, 256, and 320 produce comparable results. It suggests that those four window sizes can be used to binarize images, particularly for old Sundanese palm leaf manuscripts. However, we can see that a window size of 320 provides the most stable performance, with F-measure greater than 68 for all number of filters.

Furthermore, the highest measurement score on this experiment is obtained for a window size of 320 and 96 filters, with a F-measure of 68.44. Another important consideration is that using a larger window requires more storage and has an impact on computation costs. Afterward, the kernel size is evaluated using the previous best configuration. As shown in Fig. 4, three kernel sizes (3, 5, and 7)

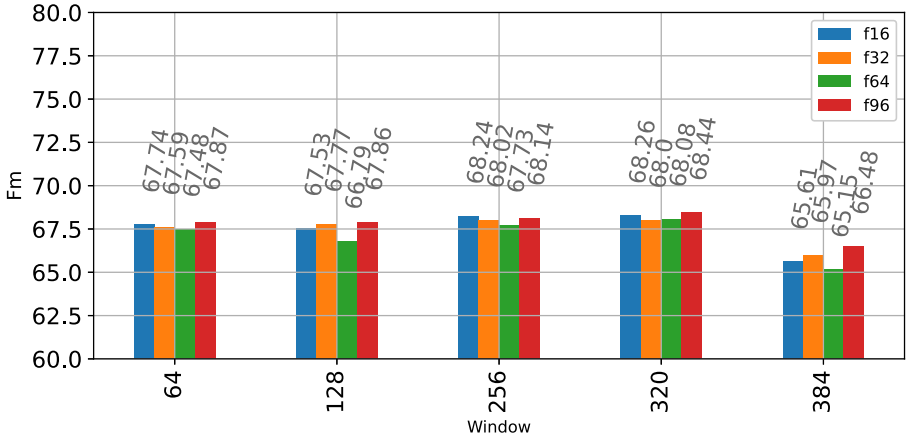

Fig. 3. The binarization performance of autoencoder network based on the number of filters across window size.

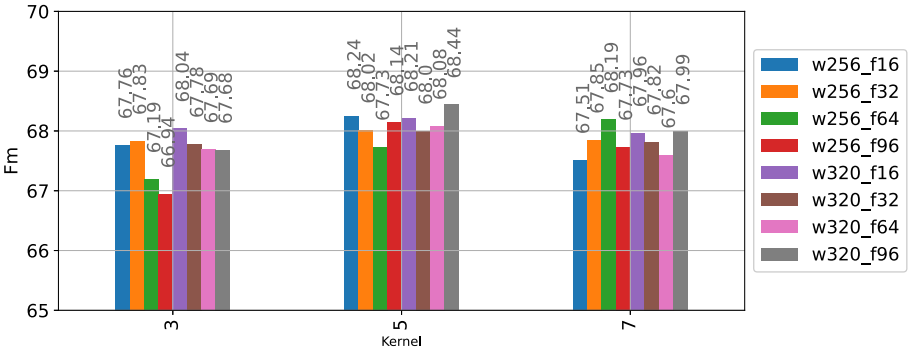

Fig. 4. The binarization performance of autoencoder network based on the number of filters across kernel.

are evaluated in eight CAE models by combining two window sizes (256 and 320) and four number of filters (16, 32, 64, and 96). The impact of kernel size varies depending on the size of the window and the number of filters. The best performance is reached with a window size of 320, 96 filters, and a kernel size of 5, which is the same as in the previous experiment.

The number of encoder-decoder blocks are evaluated in the following experiment. We reevaluated the best model from the previous experiment and a baseline configuration by varying the number of encoder and decoder blocks. As shown in Fig. 5, the performance rate fluctuates as the encoder-decoder block is added. Nonetheless, the strided convolution layer within the encoder and decoder blocks must be considered. In a configuration with a squared window size of 320 and strided convolution of 2, for example, we cannot use more than 6 encoder blocks (blue line). Because the dimension of the encoded representation from

the last encoder and the dimension of the decoded representation from the first decoder are not the same. This indicates that we must set this parameter with care.

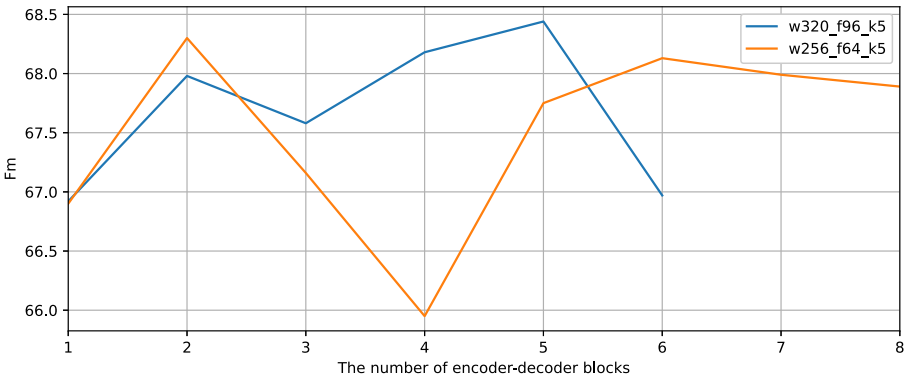

Fig. 5. The binarization performance of autoencoder network based on the number of encoder and decoder blocks. (Color figure online)

Table 1 shows how the presence of multiple convolution layers in an autoencoder network affects computation cost and performance rate. With each additional layer, the computation requires more storage to handle larger trainable and non-trainable parameters. We did not consider storage costs in this study, instead focusing on the impact of multiple convolution layers in binarization performance. We only evaluate multiple convolution layer (MCL) in CAE model with window size of 256, 64 filters, and a kernel size of 5 which is the default configuration proposed in [15]. The use of MCL may improve accuracy, but it requires more computation time and storage. According to the results of this experiment, 15 MCL in the encoder block and 15 MCL in the decoder block performed better than other settings.

Table 1. The binarization performance of autoencoder network (W256_f64_k5) based on the number of multiple convolution layers.

MCL in encoder	MCL in decoder	Trainable params	Non-trainable params	Fm
5	5	926,721	1,280	67.73
10	10	1,952,641	2,560	68.57
15	15	2,978,561	3,840	69.58
20	20	4,004,481	5,120	69.36
25	25	5,030,401	6,400	69.45

4.2 Experiments on Batch Normalization, Dropout, Skip-Connection in CAE Network

To assess the impact of the three supporting components, we set several static hyper-parameters as follows: 5 encoder/decoder blocks, each with one convolution layer and a kernel size of 5. Figure 6 and Fig. 7 depict the types of accuracy performance measures performed by CAE networks with various supporting components, covering a dropout rate range of 0 to 0.5 in the binarization task of old Sundanese palm leaf manuscripts. We can see that the accuracy of the CAE networks increases in general. The configuration with added skip connection and no batch normalization provides the best performance overall if many filters are occupied.

First, a discussion of dropout regularization is presented. Based on 4 CAE models, we discovered that the existence of dropout layer relatively increases the accuracy rate. Without dropout layer (dropout = 0.0), the F-measure reaches most of the time the lowest level for a given CAE model. On contrary, the dropout layer in range 0.2 and 0.5 generally may obtain the better accuracy. In order to obtain a global optimum performance, dropout rate could be different for each model. This conclusion corresponds to the suggestion made by the authors who proposed the dropout concept [30]. It means that dropout rate needs to be set carefully.

In terms of skip-connection, CAE models with no residual blocks cannot outperform CAE models with either added or concatenated skip-connection. The highest accuracy score for a CAE model with a residual block is based on a window size of 320×320, 96 filters, a kernel size of 5, and a dropout rate of

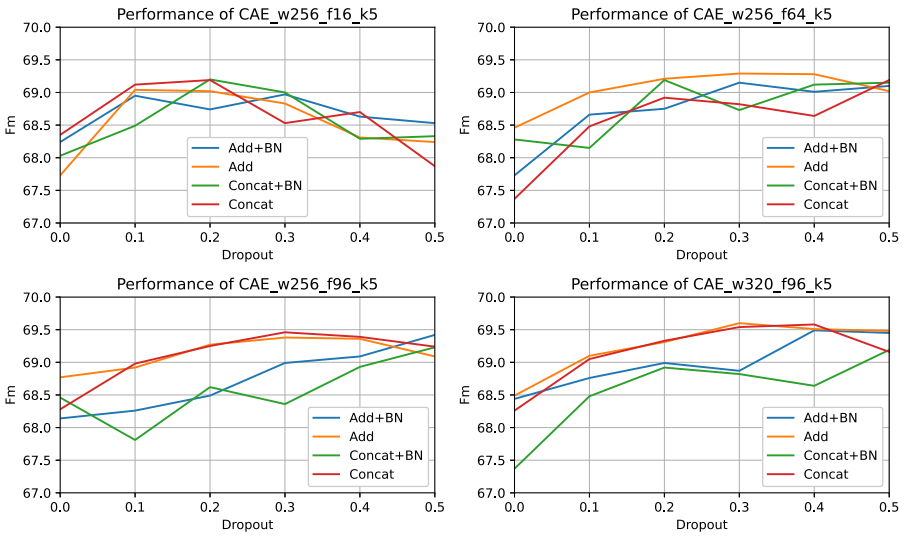

Fig. 6. An accuracy performance comparison of different dropout rates, skip connection functions and batch normalization on CAE network.

0.3. In many cases, such as the number filters of 96, the summation function (blue and orange lines) outperforms the concatenation function (green line and red line). We recommend using the residual block with the summation function. Another important point is that transferring meaningful information from the previous layer via the skip method may improve accuracy. Figure 6 depicts the performance of a residual CAE network with an accuracy rate higher than 67. In contrast, as shown in Fig. 7, a CAE model with a non-residual block can accurately segment below this rate.

As shown in Fig. 7, the BN has a significant impact on the CAE network with no skip-connection. On all dropout rates, BN within non-residual CAE (blue line) consistently outperforms the configuration without BN (orange line). Additionally, the model of residual CAE with BN and 50% dropout after each convolution block performs slightly better than the model without BN, as depicted in Fig. 6. In contrast, the residual CAE with a dropout rate of 0.4 or less in some configuration produces less performance than the model without BN. Meanwhile, the model without residual block and batch normalization fails to segment foreground and background because the F-measure is most of the time lower than 55%.

Figure 8 illustrates the qualitative results of several binarization methods applied on testing dataset of old Sundanese palm leaf manuscripts. Several experiments on supporting components in CAE architecture revealed that the highest accuracy rate (F-measure) is 69.60 for a 320×320 window size, 96 filters, 5×5 kernel, with added skip-connection, no batch normalization, and dropout rate of 0.3. Another intriguing aspect is the possibility of improving accuracy by using multiple convolutional layers. The accuracy improvement of CAE_add_BN_w256_f64_k5

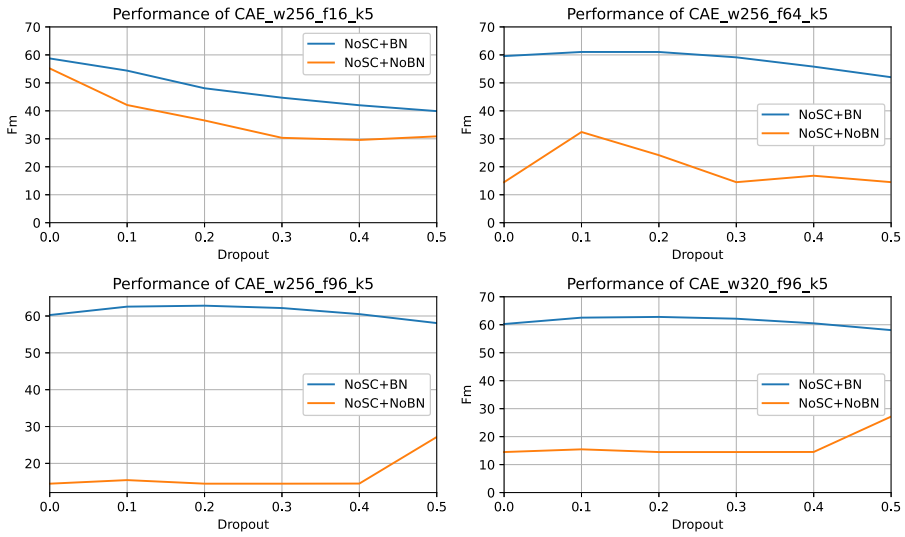

Fig. 7. A comparison of the dropout and batch normalization (BN) influence on CAE network without skip-connection (SC). (Color figure online)

model can be seen in the configuration of CAE_add_BN_30MCL_w256_f64_k5. Overall, the evaluation metric of the CAE model also validates the promising results and outperforms the winner of ICFHR 2018 [36]. Moreover, from the image visualization of a sample image, the binarized images produced by CAE models are cleaner than prior state of the art approaches in term of separate foreground from random distributed noise, as shown in Fig. 9.

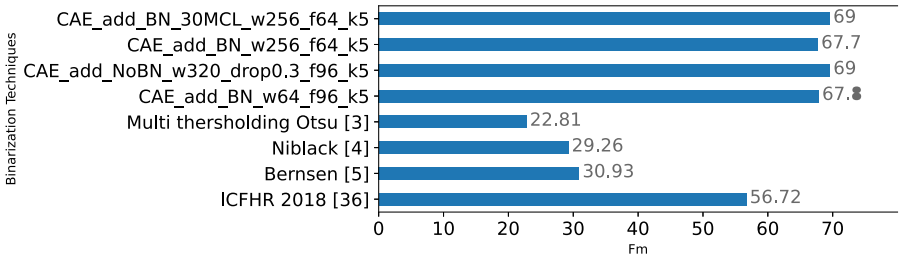

Fig. 8. Evaluation metric on several binarized images.

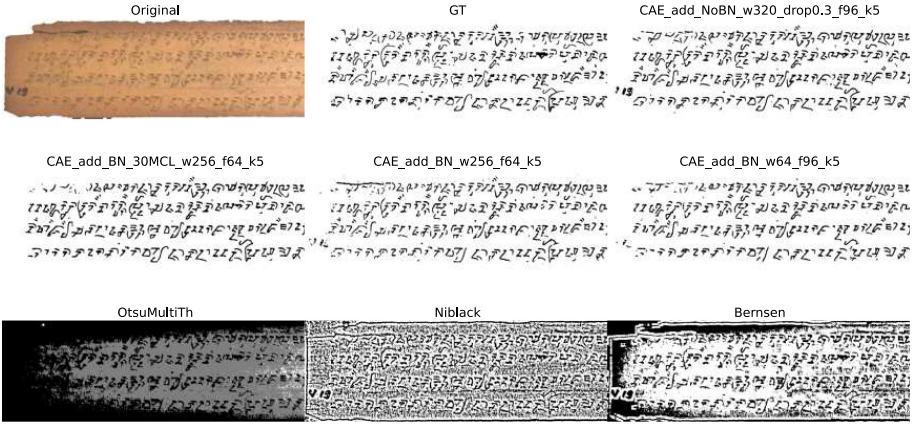

Fig. 9. A sample of original palm leaf manuscript image, ground truth image (GT) and the binarization results.

5 Conclusion

Several experiments on the CAE network's hyper-parameters and three supporting components were carried out for the binarization task on old Sundanese palm leaf manuscripts. Tuning the parameters can improve the performance of the binarization. Nevertheless, some parameters such as kernel size, number of filters and the number of encoder-decoder blocks, don't play a significant role

on accuracy measurement. However, in this network, window size and the multiple convolution layer have a significant impact. Furthermore, skip-connection of both summation and concatenation functions that supply the encoded representation from the previous layer can increase accuracy. To handle overfitting in deep network, dropout layer needs to be set up in range 0.2 and 0.5. Moreover, the batch normalization also provides a big impact if there is no residual block in CAE network. Therefore, to achieve good performance, we need to choose carefully the hyper-parameters and the supporting components. These experimental results will help us for future works on document image analysis, specifically for evaluating the CAE network with larger datasets of palm leaf manuscripts.

Acknowledgments. We would like to thank Riki Nawawi and Undang Darsa as Sundanese philologists for their fruitful discussions regarding old Sundanese manuscripts. This work is supported by the Directorate General of Higher Education, Ministry of Education, Culture, Research and Technology, the Republic of Indonesia through the funding of the BPPLN-DIKTI Indonesian Scholarship Program.

References

1. Sulaiman, A., Omar, K., Nasrudin, M.F.: Degraded historical document binarization: a review on issues, challenges, techniques, and future directions. J. Imaging **5**(4), 48 (2019)
2. Tensmeyer, C., Martinez, T.: Historical document image binarization: a review. SN Comput. Sci. **1**(3), 1–26 (2020). https://doi.org/10.1007/s42979-020-00176-1
3. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)
4. Niblack, W.: An Introduction to Digital Image Processing. Prentice-Hall, Englewood Cliffs (1986)
5. Bernsen, J.: Dynamic thresholding of gray level image. In: ICPR 1986: Proceedings of International Conference on Pattern Recognition, Berlin, pp. 1251–1255 (1986)
6. Kesiman, M.W.A., et al.: Benchmarking of document image analysis tasks for palm leaf manuscripts from Southeast Asia. J. Imaging **4**(2), 43 (2018)
7. Howe, N.R.: Document binarization with automatic parameter tuning. Int. J. Doc. Anal. Recognit. **16**(3), 247–258 (2013)
8. Mesquita, R.G., Silva, R.M.A., Mello, C.A.B., Miranda, P.B.C.: Parameter tuning for document image binarization using a racing algorithm. Expert Syst. Appl. **42**(5), 2593–2603 (2015)
9. Sauvola, J., Seppanen, T., Haapakoski, S., Pietikainen, M.: Adaptive document binarization. In: Proceedings of the Fourth ICDAR, pp. 147–152 (1997)
10. Pastor-Pellicer, J., España-Boquera, S., Zamora-Martínez, F., Afzal, M.Z., Castro-Bleda, M.J.: Insights on the use of convolutional neural networks for document image binarization. In: Rojas, I., Joya, G., Catala, A. (eds.) IWANN 2015. LNCS, vol. 9095, pp. 115–126. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19222-2_10
11. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3431–3440 (2015)
12. Shelhamer, E., Long, J., Darrell, T.: Fully convolutional networks for semantic segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(4), 640–651 (2017)

13. Tensmeyer, C., Martinez, T.: Document image binarization with fully convolutional neural networks. In: 2017 14th IAPR ICDAR, pp. 99–104 (2017)
14. Peng, X., Cao, H., Natarajan, P.: Using convolutional encoder-decoder for document image binarization. In: 2017 14th IAPR ICDAR, pp. 708–713 (2017)
15. Calvo-Zaragoza, J., Gallego, A.J.: A selectional auto-encoder approach for document image binarization. Pattern Recognit. **86**, 37–47 (2019)
16. Lazzara, G., Levillain, R., Geraud, T., Jacquelet, Y., Marquegnies, J., Crepin-Leblond, A.: The SCRIBO module of the Olena platform: a free software framework for document image analysis. In: 2011 ICDAR, pp. 252–258 (2011)
17. Pratikakis, I., Zagoris, K., Barlas, G., Gatos, B.: ICFHR 2016 handwritten document image binarization contest (H-DIBCO 2016). In: Proceedings of the International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 619–623 (2016)
18. Burie, J.C., et al.: ICFHR2016 competition on the analysis of handwritten text in images of balinese palm leaf manuscripts. In: 2016 15th ICFHR, pp. 596–601 (2016)
19. Géron, A.: Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow?: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly, Sebastopol (2019)
20. Feng, X., Jonathan, W., Q.M., Yang, Y., Cao, L.: An autuencoder-based data augmentation strategy for generalization improvement of DCNNs. Neurocomputing **402**, 283–297 (2020)
21. Ribeiro, M., Lazzaretti, A.E., Lopes, H.S.: A study of deep convolutional auto-encoders for anomaly detection in videos. Pattern Recognit. Lett. **105**, 13–22 (2018)
22. Heger, J., Desai, G., Zein El Abdine, M.: Anomaly detection in formed sheet metals using convolutional autoencoders. In: Procedia CIRP, pp. 1281–1285. Elsevier B.V. (2020)
23. Mao, X.-J., Shen, C., Yang, Y.-B.: Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In: Proceedings of the 30th International Conference on Neural Information Processing Systems, pp. 2810–2818. Curran Associates Inc., Red Hook (2016)
24. Xia, M., Liu, X., Wong, T.T.: Invertible grayscale. ACM Trans. Graph. **37**, 1–10 (2018)
25. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: 2nd International Conference on Learning Representations, ICLR 2014, Canada, pp. 1–14 (2014)
26. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: the all convolutional net. In: ICLR-2015 (2015)
27. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016)
28. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on CVPR, pp. 2261–2269 (2017)
29. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on CVPR, pp. 2818–2826 (2016)
30. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014)

31. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: The 32nd International Conference on Machine Learning, Lille, France (2015)
32. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-ResNet and the impact of residual connections on learning. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, pp. 4278–4284. AAAI Press (2017)
33. Sun, S., Pang, J., Shi, J., Yi, S., Ouyang, W.: FishNet: a versatile backbone for image, region, and pixel level prediction. In: 32nd Conference on Neural Information Processing Systems, pp. 754–764 (2018)
34. Thakkar, V., Tewary, S., Chakraborty, C.: Batch normalization in convolutional neural networks - a comparative study with CIFAR-10 data. In: 2018 Fifth International Conference on Emerging Applications of Information Technology, pp. 1–5 (2018)
35. Garbin, C., Zhu, X., Marques, O.: Dropout vs. batch normalization: an empirical study of their impact to deep learning. Multimed. Tools Appl. **79**, 12777–12815 (2020)
36. Kesiman, M.W.A., et al.: ICFHR 2018 competition on document image analysis tasks for southeast Asian palm leaf manuscripts. In: Proceedings of ICFHR, pp. 483–488. IEEE (2018)
37. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015)
38. Pastor-Pellicer, J., Zamora-Martínez, F., España-Boquera, S., Castro-Bleda, M.J.: F-measure as the error function to train neural networks. In: Rojas, I., Joya, G., Gabestany, J. (eds.) IWANN 2013. LNCS, vol. 7902, pp. 376–384. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38679-4_37