



A New Efficient Finger-Vein Verification Based on Lightweight Neural Network Using Multiple Schemes

Haocong Zheng^{1,2}, Yongjian Hu^{1,2(✉)}, Beibei Liu¹, Guang Chen^{1,3},
and Alex C. Kot⁴

¹ South China University of Technology, Guangzhou 510640, Guangdong, China
eeyjhu@scut.edu.cn

² Sino-Singapore International Joint Research Institute, Guangzhou 10700,
Guangdong, China

³ GRG Banking Equipment Co., Ltd., Guangzhou 510663, Guangdong, China

⁴ Rapid-Rich Object Search (ROSE) Lab, Nanyang Technological University,
Singapore 639798, Singapore

Abstract. Existing deep learning-based finger-vein algorithms tend to use large-scale neural networks. From the perspective of computational complexity, this is not conducive to practical applications. Besides, in our opinion, finger-vein images often have relatively simple textures and are small in image size, it is not economical to use large-scale neural networks. Inspired by the increasing accuracy of lightweight neural networks on ImageNet, we introduce the lightweight neural network ShuffleNet V2 as a backbone to construct a basic pipeline for finger-vein verification. To customize the network for this application, we propose schemes to improve it from the aspects including data input, network structure, and loss function design. Experimental results on three public databases have exhibited the excellence of the proposed model.

Keywords: Finger-vein verification · Lightweight neural network · ShuffleNet V2 · Data augmentation · Joint loss function

1 Introduction

Biometric features are widely used for authentication and identification applications. The most commonly used ones are the fingerprint, human face, voice, iris and finger-vein. Compared with other biometric features, the finger-vein has some prominent advantages. For example, the finger-vein is an internal biometric modality, and thus is much harder to copy and forge. On the other hand, the acquisition of finger-vein images is quick and friendly.

Supported in part by Sino-Singapore International Joint Research Institute (No. 206-A017023, No. 206-A018001), Science and Technology Foundation of Guangzhou Huangpu Development District under Grant 201902010028, and NTU-PKU JRI.

© Springer Nature Switzerland AG 2020

I. Farkaš et al. (Eds.): ICANN 2020, LNCS 12396, pp. 748–758, 2020.

https://doi.org/10.1007/978-3-030-61609-0_59

However, there are also some challenges for finger-vein verification. For example, the captured finger-vein images usually have low contrast and are easy to be affected by uneven illumination, temperature changes, 2D or 3D rotation, and other noise. Low-quality finger-vein images are difficult to classify correctly by employing manual feature extraction and matching methods. Compared with hand-crafted features, deep learning features extracted by convolutional neural networks (CNN) have proven to be more general and representative, and are being used in various computer vision tasks.

To use deep learning features for finger-vein verification, one needs to train the convolutional neural network with a huge amount of training data. Unfortunately, the size of public finger-vein databases is relatively small. As a result, the issue of creating training samples through data augmentation has attracted much attention. It is obvious that good recognition performance and satisfactory robustness can only be achieved by a deliberately designed and well trained neural network. Recent finger-vein studies mainly discuss these problems. For industrial applications, the computational complexity is also highly concerned.

Qiu et al. [1] proposed a region of interest (ROI) extraction method which is robust to light illumination and 3D rotation, but the performance is not satisfactory. Qin et al. [2] first designed a CNN to extract the vein patterns from any image region, and then recovered the missing finger-vein patterns in the binary image based on a fully convolutional network. They employed a template matching method for finger-vein verification. The whole process is too complex and time-consuming. Fang et al. [3] proposed a two-channel network to tackle the problem of lack of finger-vein data, and a two-stream network to overcome image displacement. The above two types of networks are integrated with a selective network to achieve good performance. Hu et al. [4] proposed the Finger-vein Network (FV-Net) to learn more discriminative features of finger-vein and addressed the misalignment problem like translation and rotation in vein imaging. Zeng et al. [5] proposed a new fully convolutional neural network (FCN) and integrated it with conditional random field to extract vein textures. The FCN takes the U-Net [6] as its basic structure and introduces deformable convolution and residual recurrent convolution to extract and retain deeper and more complex features. Hao et al. [7] proposed a multi-task neural network model algorithm to jointly carry out the ROI extraction task and the feature extraction task. By combining these two tasks, the quality of extracted vein features is improved.

In this article, we propose a lightweight deep learning model for finger-vein verification. We introduce a data augmentation scheme to mitigate the pressure of lack of training samples with brightness weakness, being partially cropped or rotated. We further modify a lightweight ShuffleNet V2 [8] to extract a more efficient feature map with a larger size. The label smoothing [9] and the joint loss function from [10] are also introduced for learning discriminative features. Experiments on databases including SDUMLA-HMT [11], FV-USM [12], and MMCBNU_6000 [13] have demonstrated that the proposed method is efficient and outperforms five current methods in literature.

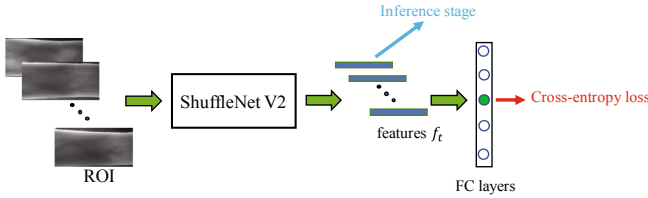
2 The Basic Framework for Finger-Vein Verification

2.1 The Baseline

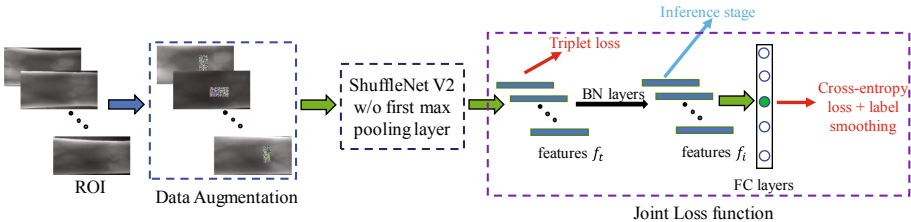
The pipeline of our basic model is shown in Fig. 1(a). We introduce the ShuffleNet V2 as the backbone network for our finger-vein verification task. In the training stage, we train the network with cross-entropy loss. In the inference stage, we compare the images by using the cosine similarity of features learned from the network. The cross-entropy loss is shown in Eq. (1).

$$L_S = - \sum_{i=1}^M p_i \log q_i \tag{1}$$

Where p is the truth probability distribution, q is the prediction probability distribution and M is the number of categories. Assume that the true label of the image is y . We let $p_i = 1$ if $i = y$ and $p_i = 0$ otherwise.



(a) Basic pipeline



(b) Modified pipeline after using all schemes

Fig. 1. Basic pipeline and our proposed framework

2.2 Training and Inference

The training procedure employed is listed as follows.

1. We set the number of epochs to 500 and the batch size to 128. For each epoch, we randomly shuffle the images and normalize pixel value to $[0, 1]$ as the input.

2. Finger-vein images are fed into the network to get the feature f_t . We obtain the prediction probability distribution by passing f_t through the fully connected layer and softmax function. After that, we calculate the cross-entropy loss based on the prediction probability distribution and the truth probability distribution, and update loss through backpropagation.
3. We adopt Adam optimizer and use Eq. (2) to determine the learning rate, where t is the number of epochs. We first adopt a gradual warmup strategy [14] to linearly increase the learning rate from 0 to the initial learning rate. Because at the beginning of network training, the network parameters are far away from the final solution. In this situation, using a large learning rate may result in model instability. With a warmup strategy, the model can become more stable. During training deep neural networks, it is usually helpful to drop the learning rate over time. Therefore we use a step decay learning rate schedule that drops the learning rate by a factor according to some predefined steps. The factor and the predefined steps are determined by manual tuning.

$$lr(t) = \begin{cases} \frac{t}{10} \times 3.5 \times 10^{-3}, & t \leq 10 \\ 3.5 \times 10^{-3}, & 10 < t \leq 180 \\ 7 \times 10^{-4}, & 180 < t \leq 300 \\ 1.4 \times 10^{-4}, & 300 < t \leq 360 \\ 2.8 \times 10^{-5}, & 360 < t \leq 500 \end{cases} \quad (2)$$

In our inference procedure, we normalize the input image to the value interval $[0, 1]$, and feed the normalized data to the trained network to get the features. During finger-vein verification, the cosine similarity between features is used directly for comparison.

3 Our Improvement Schemes

The basic model proposed above is an end-to-end algorithm. It is simple but the performance is not good. We improve it from several aspects including input data augmentation, network structure, and loss function design.

3.1 Input Data Augmentation

Deep learning model often needs a large training set. Unfortunately, there are not many databases available for finger-vein. For public finger-vein databases like SDUMLA-HMT, FV-USM, and MMCBNU_6000, the size of databases is still small. Meanwhile, some finger-vein images suffer from uneven brightness, missing textures, cropping, translation, rotation, and so on. To let the model learn about the invariance of vein pattern, we present a data augmentation scheme to simulate sample generation. The overall data augmentation pipeline is shown in Fig. 2.

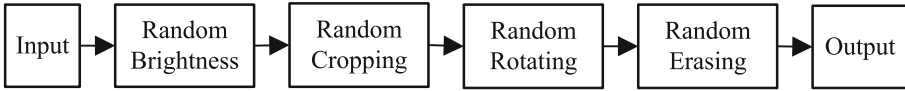


Fig. 2. Data augmentation pipeline

During the model training, the brightness of each input ROI image is randomly adjusted with a factor in a range of 0.7–1.3. Then, the ROI image is randomly cropped with a crop box of 0.9 times the original image size and resized back to the original image size. Next, the ROI image is randomly rotated by an angle between -3 and 3° . Finally, random erasing [15] is adopted. It can be seen that this data augmentation pipeline can greatly increase the number and varieties of training samples, making the model more robust against the mentioned disadvantages.

3.2 Network Improvement

ShuffleNet V2 network has been proven to achieve excellent performance in multiple computer vision tasks, including classification and object detection tasks. And the ShuffleNet V2 network is lightweight and efficient, so we take this network as the basic structure. However, the ShuffleNet V2 network was originally designed for 224×224 input images. If it is directly applied to the 128×64 ROI image, the output feature maps only have the size of 4×2 . Apparently, that size is too small for the extraction of detailed features. Considering that a pooling layer would cause information missing, we remove the first max pooling layer so as to generate the feature map of size 8×4 . This modification seems simple and straightforward, but the effect on performance improvement is enormous, as will be seen later. In fact, the output feature map with a larger size can better preserve the fine-grained features, which is especially important for finger-vein images. The modified network structure is shown in Table 1.

For model training, one serious problem would be raised due to lack of finger-vein training samples. If the weights are initialized by random initialization like Xavier, the network would hardly learn discriminative features from the limited data, and the final model would perform poorly. Therefore, we use the transfer learning concept and pre-train our backbone network on the ImageNet dataset. ImageNet is a large dataset that has a wide variety of objects. The massive data set ensures that the pre-trained network provides high generalization. Particularly, the shallow features learned by the pre-trained network, such as edges, textures, and curves, have excellent versatility. These shallow features also play an important role in finger-vein verification. Therefore, the pre-trained model can provide a useful starting point for the finger-vein verification task. The weights obtained are saved as the initial weights.

Table 1. The modified ShuffleNet V2 structure. The stage layer is constructed by repeatedly stacking the spatial down sampling unit and the basic unit proposed in [8].

Layer	Output size	Kernel size	Stride	Repeat	Output channels
Image	128×64	–	–	–	3
Conv1	64×32	3×3	2	1	24
Stage2	32×16 32×16	–	2 1	1 7	116
Stage3	16×8 16×8	–	2 1	1 7	232
Stage4	8×4 8×4	–	2 1	1 3	464
Conv5	8×4	1×1	1	1	1024
GlobaPool	1×1	8×4	–	–	–

3.3 Label Smoothing and Loss Function

We adopt the label smoothing to reduce overfitting. It changes the truth probability distribution to l_i as shown in Eq. (3), where ε is a small constant to encourage the model to be less confident on training set. Empirically, ε is set to be 0.1.

$$l_i = \begin{cases} \varepsilon/(M-1), & i \neq y \\ 1 - \varepsilon, & i = y \end{cases} \quad (3)$$

We define the loss after adding label smoothing L_{Sl} as follows

$$L_{Sl} = - \sum_{i=1}^M l_i \log q_i \quad (4)$$

The cross-entropy loss is suitable for feature separation of different categories but does not consider intra-class feature gathering. The learned features which are not compact within the intra class would reduce the performance of our finger-vein verification. On the other hand, the widely used triplet loss can effectively enhance the intra-class compactness in Euclidean space but can not provide globally optimal constraint. It would be more conducive for model training if these two loss functions are combined. However, the learning targets of these two loss functions are inconsistent. Cross-entropy loss mainly optimizes the cosine distance while triplet loss mainly optimizes the Euclidean distances. Luo et al. [10] proposed a simple way that added a batch normalization (BN) layer after features to solve the problem. It can help reduce the constraint between cross-entropy loss and triplet loss. Therefore, more discriminative features can be learned with the joint loss function. The triplet loss is shown in Eq. (5).

$$L_{Tri} = [d_p - d_n + \alpha]_+ \quad (5)$$

Where d_p and d_n are the Euclidean distances between positive and negative pairs. α is the margin of triplet loss. Empirically, α is set to 0.3. The triplet selection scheme is to randomly select P fingers, and each finger randomly selects K images to form a batch. For each image in the batch, it selects one image from different classes with the smallest Euclidean distance of the corresponding features and another image of the same class with the largest Euclidean distance to form a triplet. In our work, P is set to 32 and K to 4. The total loss function is $L = L_{SI} + L_{Tri}$.

The modified pipeline is shown in Fig. 1(b). In the inference stage, the batch normalized feature f_i is used to measure the similarity.

4 Experiment and Analysis

In the training stage, we train the model on Ubuntu 18.04 system with a CPU E5-2683 v3@2.00 GHz and an NVIDIA Corporation GP102 (TITAN XP). In the test stage, the trained model runs on a Windows 10 system with a Ryzen 5 2500U@2.00 GHz. We conduct experiments on three public finger-vein databases which are SDUMLA-HMT, FV-USM, and MMCBNU_6000. The outline of the three databases are given in Table 2. Note that the ROI images are extracted using the method similar to the literature [16] to extract the ROI image. The difference is that we crop the outer rectangle of the whole finger area as the ROI and resize to 128×64 . The extracted ROIs are shown in the second line of Fig. 3.

For the division of dataset, we randomly choose one half of the classes for training, and the remaining is only used for testing. For each class in the test set, half of the images are randomly selected as the gallery, and the remaining half is used as the probes. The probes and the samples in the gallery are used to construct the positive and negative sample pairs. We use the equal error rate (EER) to evaluate the performance of our finger-vein verification model. The EER is the ratio when the false acceptance rate and false rejection rate are equivalent. The lower the EER, the better the verification performance.

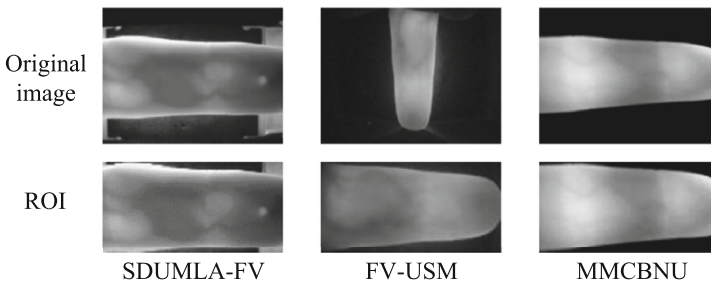


Fig. 3. Example images and their extracted ROIs from the three finger-vein databases

Table 2. The outline of three Finger-vein Databases

Databases	Subjects	Fingers per subject	Images per finger	Image size	Total images
SDUMLA-HMT	106	6	6	320 × 240	3816
FV-USM	123	4	12	640 × 480	5904
MMCBNU_6000	100	6	10	640 × 480	6000

4.1 Effect of Our Schemes on Model Performance

We use an ablation study to analyze the effect of our schemes on the input, network, and loss function. The results are shown in Table 3. Our data augmentation and the joint loss function can train the model well so it performs better than the baseline model. Next, the modified network further decreases the EERs due to the output of larger feature maps. Moreover, by using the pre-training, our model can further improve the performance. The additional use of label smoothing and joint loss function can make our model achieve the lowest EERs in all three databases. In summary, after applying all the schemes, the EERs of our finger-vein verification model decrease from 6.59%, 2.24%, 2.03% to 0.37%, 0.31%, and 0.05% in the order of SDUMLA-HMT, FV-USM, and MMCBNU_6000.

Table 3. Effect of our schemes on model performance (EER%)

Model	SDUMLA-HMT	FV-USM	MMCBNU_6000
Baseline	6.59	2.24	2.03
+Data augmentation	3.19	1.59	0.85
+Modified network	2.72	0.84	0.39
+Pre-trained	0.96	0.63	0.33
+Label smoothing	0.77	0.54	0.15
+Joint loss function	0.37	0.31	0.05

4.2 Effect of Backbone Network on Model Performance

We analyze the effect of different backbone networks on model performance. To investigate the effect in detail, we replace the backbone network with MobileNet V2 [17], ResNet18 [18], ResNet50 [18], and DenseNet121 [19] one by one under the same schemes such as data augmentation, ImageNet pretraining, label smoothing and joint loss function. As shown in Table 4, DenseNet121 and the modified ShuffleNet V2 achieve better results than the other networks. It is because both networks use a feature reuse structure, which can combine the advantages of low-level features and high-level features. Between them, our modified ShuffleNet V2 can better suit the finger-vein input size, so it gets the best results in the three public databases.

Table 4. Effect of different backbone networks on model performance (EER%)

Backbone	SDUMLA-HMT	FV-USM	MMCBNU_6000
MobileNet V2	0.82	0.48	0.21
ResNet-18	1.21	0.52	0.5
ResNet-50	0.80	0.45	0.21
DenseNet-121	0.73	0.33	0.14
Modified ShuffleNet V2	0.37	0.31	0.05

4.3 Comparison with Other Methods

The classical machine learning-based method DWP [1], and four state-of-the-art deep learning-based methods, Selective-network [3], FV-Net [4], FCN [5] and MTL [7] are selected for comparison. Tables 5 and 6 show that DWP runs the fastest in feature extraction and matching, but its performance is the worst among 6 methods. Our method is mainly compared with the four deep learning-based methods. FV-Net uses a modified VGGFace-Net as the backbone and the FLOPs (floating-point operations) are about 51 times of ours. Selective-network uses a lightweight neural network, but it needs to extract the mini-ROI as an additional input, and the average time spent on mini-ROI extraction is 50ms. FCN introduces deformable convolution and recursive convolution in the U-Net structure, which will result in some computational cost. And FCN uses multiple overlapping patches for prediction, which increases the FLOPs, it is the most time-consuming among all methods. MTL takes the original image as input. Since the original image has a large size, it will bring additional computational cost. Note that the FCN method and the MTL method have different input sizes for different databases, so the FLOPs are not fixed. It can be seen that our method can reach the lowest EERs in all three databases. In terms of FLOPs, the number of weights, and the average feature extracting time, our model performs the best. As for the average matching time, our model is slightly slower than Selective-network and MTL. On the whole, our model has the best overall performance.

Table 5. EER (%) of different methods

Methods	SDUMLA-HMT	FV-USM	MMCBNU_6000
DWP [1]	1.59	2.78	2.32
Selective-network [3]	0.47	–	0.30
FV-Net [4]	1.20	0.76	0.30
FCN [5]	5.827	–	0.364
MTL [7]	1.17	0.74	0.29
Proposed method	0.37	0.31	0.05

Table 6. Detailed comparison of different methods in the test stage

Methods	FLOPs	Number of weights	Average extracting time	Average matching time
DWP [1]	–	–	0.0175 ms	0.13 ms
Selective-network [3]	119.2M	6.04M	140 ms	0.18 ms
FV-Net [4]	4.88G	3.51M	684 ms	0.98 ms
FCN [5]	0.17–2.67T	193.22M	217 s	630 ms
MTL [7]	2.04–6.94G	13.16M	121 ms	0.19 ms
Proposed method	95.4M	1.26M	97 ms	0.20 ms

5 Conclusion

In this article, we have presented a method of using the lightweight neural network for finger-vein verification. We first introduced a baseline network, and then discussed the schemes of how to improve the model from several aspects including data augmentation, network modification and loss function design. The experiments validate the proposed framework. The test results on the three public databases have demonstrated that our method can significantly outperform one traditional method and well outperform four state-of-the-art CNN-based methods in terms of verification performance. Meanwhile, it needs less FLOPs and smaller memory than the four CNN-based methods.

References

1. Qiu, S., Liu, Y., Zhou, Y., Huang, J., Nie, Y.: Finger-vein recognition based on dual-sliding window localization and pseudo-elliptical transformer. *Expert Syst. Appl.* **64**, 618–632 (2016)
2. Qin, H., El-Yacoubi, M.A.: Deep representation-based feature extraction and recovering for finger-vein verification. *IEEE Trans. Inf. Forensics Secur.* **12**(8), 1816–1829 (2017)
3. Fang, Y., Wu, Q., Kang, W.: A novel finger-vein verification system based on two-stream convolutional network learning. *Neurocomputing* **290**, 100–107 (2018)
4. Hu, H., Kang, W., Lu, Y., Fang, Y., Liu, H., Zhao, J., Deng, F.: FV-Net: learning a finger-vein feature representation based on a CNN. In: 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, China, pp. 3489–3494. IEEE (2018)
5. Zeng, J., et al.: Finger vein verification algorithm based on fully convolutional neural network and conditional random field. *IEEE Access* **8**, 65402–65419 (2020)
6. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) *MICCAI 2015*. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28

7. Hao, Z., Fang, P., Yang, H.: Finger vein recognition based on multi-task learning. In: Proceedings of the 2020 5th International Conference on Mathematics and Artificial Intelligence, Chengdu, China, pp. 133–140, ACM Digital (2020)
8. Ma, N., Zhang, X., Zheng, H.-T., Sun, J.: ShuffleNet V2: practical guidelines for efficient CNN architecture design. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018. LNCS, vol. 11218, pp. 122–138. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_8
9. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, Nevada, USA, pp. 2818–2826, IEEE (2020)
10. Luo, H., Gu, Y., Liao, X., Lai, S., Jiang, W.: Bag of tricks and a strong baseline for deep person re-identification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA. IEEE (2019)
11. Yin, Y., Liu, L., Sun, X.: SDUMLA-HMT: a multimodal biometric database. In: Sun, Z., Lai, J., Chen, X., Tan, T. (eds.) CCBIR 2011. LNCS, vol. 7098, pp. 260–268. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25449-9_33
12. Asaari, M.S.M., Suandi, S.A., Rosdi, B.A.: Fusion of band limited phase only correlation and width centroid contour distance for finger based biometrics. *Expert Syst. Appl.* **41**(7), 3367–3382 (2014)
13. Lu, Y., Xie, S. J., Yoon, S., Wang, Z., Park, D. S.: An available database for the re-research of finger vein recognition. In: 2013 6th International Congress on Image and Signal Processing (CISP), Hangzhou, China, vol. 1, pp. 410–415, IEEE (2013)
14. Goyal, P., et al.: Accurate, large minibatch SGD: training ImageNet in 1 hour. arXiv preprint [arXiv:1706.02677](https://arxiv.org/abs/1706.02677) (2017)
15. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. arXiv preprint [arXiv:1708.04896](https://arxiv.org/abs/1708.04896) (2017)
16. Wang, M., Tang, D.: Region of interest extraction for finger vein images with less information losses. *Multimed. Tools Appl.* 1–13 (2016). <https://doi.org/10.1007/s11042-016-4285-2>
17. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L. C.: Mobilenetv 2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, USA, pp. 4510–4520, IEEE (2018)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, Nevada, USA. pp. 770–778, IEEE (2016)
19. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K. Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii, USA, pp. 4700–4708, IEEE(2017)