Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 19(12): 13782–13798.
DOI: 10.3934/mbe.2022642
Received: 03 August 2022
Revised: 04 September 2022
Accepted: 15 September 2022
Published: 19 September 2022

Research article

Lightweight network study of leather defect segmentation with Kronecker product multipath decoding

Zhongliang Zhang[1,2], Yao Fu[1], Huiling Huang[2], Feng Rao[1,*] and Jun Han[2,*]

[1] School of Physical and Mathematical Sciences, Nanjing Tech University, Nanjing, Jiangsu 211816, China

[2] Quanzhou Institute of Equipment Manufacturing, Fujian Institute of Research on the Structure of Matter, Chinese Academy of Sciences, Quanzhou, Fujian 362201, China

* **Correspondence:** Email: raofeng2002@163.com, junhan@fjirsm.ac.cn.

Abstract: In the leather production process, defects on the leather surface are a key factor in the quality of the finished leather. Leather defect detection is an important step in the leather production process, especially for wet blue leather. To improve the efficiency and accuracy of detection, we propose a leather segmentation network using the Kronecker product for multi-path decoding and named KMDNet. The network uses Kronecker products to construct a new semantic information extraction layer named KPCL layer. The KPCL layer is added to the decoding network to form new decoding paths, and these different decoding paths are combined that segment the defective part of the leather image. We collaborate with leather companies to collect relevant leather defect images; use Tensorflow for training, validation, and testing experiments; and compare the detection results with non-machine learning algorithms and semantic segmentation algorithms. The experimental results show that KMDNet has a 1.99% improvement in $F1$ score compared to UNet for leather and a nearly three times improvement in detection speed.

Keywords: wet blue leather defect; semantic segmentation; multi-path decoding; Kronecker product; feature fusion

1. Introduction

Leather is one of the most essential raw materials for household items in people's lives today, for example, fur clothing, leather shoes, leather seats, etc. Most studies on leather defect detection algorithms are based on computer vision and mainly focus on finished leather [1–7], but there are fewer studies on wet blue leather. Compared with finished leather, wet blue leather is still in a semi-processed state, its leather is moister and its surface is relatively uneven. So the captured images of

wet blue leather are also more likely to have uneven light and dark and are more difficult to detect.

Depending on the detection effect, leather defect detection algorithms are mainly classified into classification and segmentation. Classification algorithms detect defects in the form of square wireframes and include principal component analysis [8], support vector machines [9, 10], and multilayer perceptrons [11, 12]. The segmentation algorithm is used to segment the defects based on the edges of the defective part so that it can calculate the exact utilization of wet blue leather. This algorithm includes frequency analysis [13], statistics [14, 15], modeling methods [16], and deep learning methods.

For defects with black lines and wrinkles, the classification and segmentation detection algorithms are proposed in [17]. In which, the AlexNet is used for the image classification algorithm and the U-Net is used for the segmentation algorithm. They achieved a 95% classification detection performance and a 99.84% crossover rate in their experiments with 250 defective samples and 125 non-defective samples. In [18], the authors presented a texture defect detection algorithm based on a deep convolutional generative adversarial network, which reconstructs the input image so that it is different from the original input image to achieve initial defect segmentation. In addition, the linear discriminant analysis (LDA) sub-module for error elimination detection has been added to the model. However, the leather detected by the algorithm belongs to the finished leather, while the wet blue leather has not been experimented with. The detection objects are mainly items with regular texture on the surface, which is not conducive to practical applications.

For the surface detection of five types of wet blue leather defects as decayed surfaces, open wounds, puncture wounds, insect bites, and rotten grains, Chen et al. [19] introduced a leather defect segmentation algorithm based on hyperspectral technology. The algorithm first uses hyperspectral to image the leather and then uses a 1D-convolutional neural network (CNN), 2D-UNet, and 3D-UNet to segment the defect areas, and the three networks have different advantages for different leather defects. Shen et al. [20] constructed a kronecker product matching (KPM) module using the Kronecker product to find the associated features by running the features of two images through the Kronecker product to achieve the character of recognition. Xiao et al. [21] constructed a new fully-connected way of a separable layer using the Kronecker product and verified that a separable layer is better than the fully-connected layer for classification by replacing the fully-connected layer in the classical classification recognition network.

Existing deep learning algorithms detect fewer types of wet blue leather defects, and there is still room for improvement in detection speed. To detect more defects and improve the detection speed, a lightweight semantic segmentation network with multi-path decoding is proposed in this paper. First, a module that can extract global semantic information from the image is constructed using the Kronecker product. Then, the module is applied between different levels of encoding networks. Finally, a multi-path decoding network is formed by upsampling. The semantic segmentation network thus constructed can perform accurate segmentation recognition of 10 different leather defects. After extensive training, a leather defect segmentation network with more satisfactory results is obtained.

The rest of this paper is arranged as follows. An introduction of related methods is presented in Section 2. Section 3 provides the network structure regarding encoding and decoding networks. The descriptions of the dataset used, the principal image processing explanation, the experimental configurations, and the experiment results are provided in Section 4. Finally, the conclusion is drawn in Section 5, accompanied by methodological recommendations for future work.

2. Methods

The Kronecker product is a theorem proposed by mathematician Leopold Kronecker and is a special form of the tensor product [22]. For any matrices $A = [a_{ij}] \in \mathbb{R}^{m_1 \times n_1}$ and $B = [b_{ij}] \in \mathbb{R}^{m_2 \times n_2}$, their Kronecker product, i.e., the direct product or tensor product, denoted as $A \otimes B$, is defined by

$$M = A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n_1}B \\ \vdots & \ddots & \vdots \\ a_{m_1 1}B & \cdots & a_{m_1 n_1}B \end{bmatrix} \tag{2.1}$$

and $M \in \mathbb{R}^{m \times n}$ ($m = m_1 m_2$, $n = n_1 n_2$) is a block matrix whose ij-th block is $a_{ij}B$. More specifically, it can be expressed as follows:

$$M = \begin{bmatrix} a_{11}b_{11} & a_{11}b_{12} & \cdots & a_{11}b_{1n_2} & \cdots & \cdots & a_{1n_1}b_{11} & a_{1n_1}b_{12} & \cdots & a_{1n_1}b_{1n_2} \\ a_{11}b_{21} & a_{11}b_{22} & \cdots & a_{11}b_{2n_2} & \cdots & \cdots & a_{1n_1}b_{21} & a_{1n_1}b_{22} & \cdots & a_{1n_1}b_{2n_2} \\ \vdots & \vdots & \ddots & \vdots & & & \vdots & \vdots & \ddots & \vdots \\ a_{11}b_{m_2 1} & a_{11}b_{m_2 2} & \cdots & a_{11}b_{m_2 n_2} & \cdots & \cdots & a_{1n_1}b_{m_2 1} & a_{1n_1}b_{m_2 2} & \cdots & a_{1n_1}b_{m_2 n_2} \\ \vdots & \vdots & & \vdots & \ddots & & \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots & & \ddots & \vdots & \vdots & & \vdots \\ a_{m_1 1}b_{11} & a_{m_1 1}b_{12} & \cdots & a_{m_1 1}b_{1n_2} & \cdots & \cdots & a_{m_1 n_1}b_{11} & a_{m_1 n_1}b_{12} & \cdots & a_{m_1 n_1}b_{1n_2} \\ a_{m_1 1}b_{21} & a_{m_1 1}b_{22} & \cdots & a_{m_1 1}b_{2n_2} & \cdots & \cdots & a_{m_1 n_1}b_{21} & a_{m_1 n_1}b_{22} & \cdots & a_{m_1 n_1}b_{2n_2} \\ \vdots & \vdots & \ddots & \vdots & & & \vdots & \vdots & \ddots & \vdots \\ a_{m_1 1}b_{m_2 1} & a_{m_1 1}b_{m_2 2} & \cdots & a_{m_1 1}b_{m_2 n_2} & \cdots & \cdots & a_{m_1 n_1}b_{m_2 1} & a_{m_1 n_1}b_{m_2 2} & \cdots & a_{m_1 n_1}b_{m_2 n_2} \end{bmatrix}. \tag{2.2}$$

The application of the Kronecker product in convolutional neural networks is initially the Kronecker convolution, which employs the Kronecker product to expand the standard convolutional kernel so that feature vectors neglected by atrous convolutions can be captured [23]. It is similar in idea to hole convolution [24]. The Kronecker convolution uses an additional transformation matrix as shown:

$$F = \begin{bmatrix} I_{r_2 \times r_2} & \\ & O_{(r_1 - r_2) \times (r_1 - r_2)} \end{bmatrix}_{r_1 \times r_1}, \quad 1 \le r_2 \le r_1, \tag{2.3}$$

where the upper left corner I is a $r_2 \times r_2$ square matrix which has all the element values of 1, and the lower right corner O is a zero matrix. r_1 is the inter-dilating factor that controls the dilation rate of the convolutions. r_2 is denoted as the intra-sharing factor to control the size of subregions to capture feature vectors and share filter vectors. Then, the kernel K of Kronecker convolution can be enlarged by computing the Kronecker product of F and K, that is, the new kernel K' can be formulated as

$$K' = K \otimes F. \tag{2.4}$$

Moreover, the use of the Kronecker product can greatly reduce the complexity of the calculation and also save computer storage space. After performing the Kronecker product with the traditional convolution kernel K, a new convolution kernel K' with an expansion factor is obtained. When convolving with the input, not only the perceptual field can be greatly expanded, but also the expansion size of the convolution kernel can be controlled.

In general, the structure for feature extraction in CNN is the convolutional layer. When extracting global semantic information, the size of the convolutional kernel and the number of parameters are often large and not easy to train. Alternatively, the fully connected layer can be used to extract global semantic information. The fully-connected layer is usually placed at the end of the network, connecting the feature map and the labels. Each pixel of a label represents the probability that the input image belongs to that location class. The fully connected layer also connects all pixels of the feature map, which means that all pixels of the feature map are used to determine the probability that the input image belongs to a certain category.

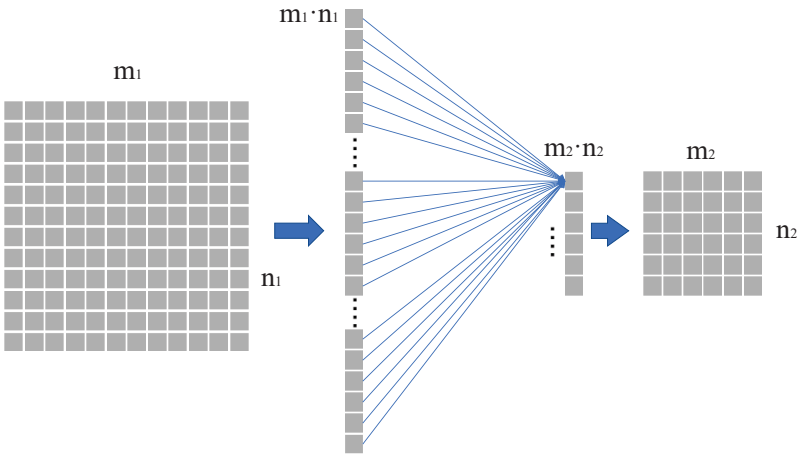


Figure 1. Full connection down-sampling.

Therefore, the extraction of global semantic information with a fully connected layer requires the idea shown in Figure 1. Each feature plane of the feature map is expanded into a one-dimensional form, and then a fully connected layer is added to extract semantic information, and the output is recombined into the form of a three-dimensional matrix. Assuming that the size of the feature map is $\mathbb{R}^{m_1 \times n_1}$, the feature map is first expanded in one dimension. Then the expanded feature map is compressed with the fully connected layer to obtain the size $\mathbb{R}^{m_2 \times n_2}$. Finally, the result is recombined into the form of a two-dimensional matrix. In this process, the number of fully connected parameters is $W = m_1 \times n_1 \times m_2 \times n_2$. If $m_1 = n_1 = 256$ and $m_2 = n_2 = 16$, the number of parameters of the fully connected layer is 10^7 levels, which is several times of the classical semantic segmentation network.

Here, we replace the parameters of the fully connected layer with two small matrices of the same size to obtain the parameters of the fully connected layer of the same size according to the Kronecker product, as shown in Figure 2, where orange indicates the original feature map and cyan is the output. The matrices A and B represent the small parameter matrices that constitute the fully connected parameters, and the operation between A and B is also the Kronecker product. This new fully connected method is called the Kronecker product connected layer (KPCL). The large parameter matrix formed by matrices A and B is multiplied with each layer of the feature map, i.e., the KPCL layer is multiplied with each layer of the feature map, and the KPCL layer does not change the position relationship of channel directions in the feature map.

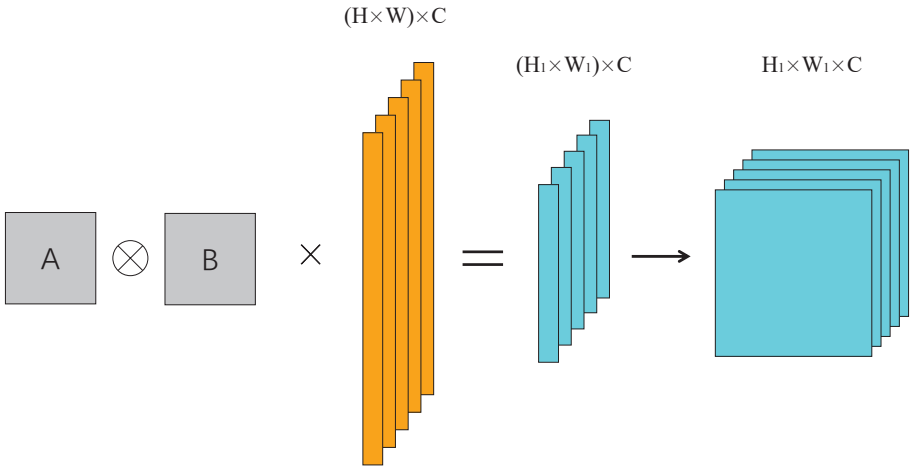


Figure 2. Schematic diagram of Kronecker product connection layer.

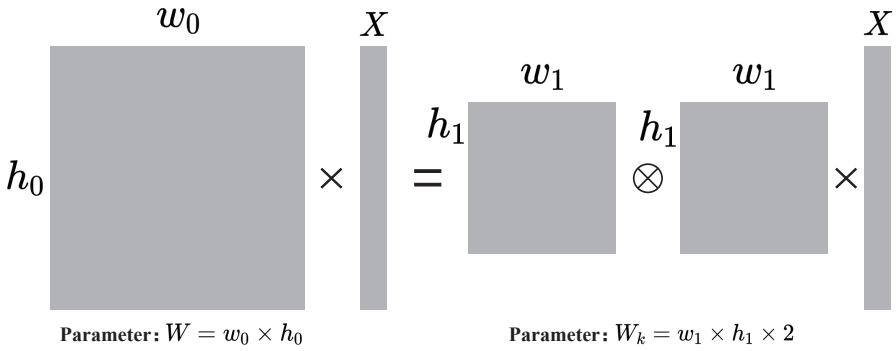


Figure 3. Schematic diagram of Kronecker product substitution parameter matrix.

In Figure 3, some details of the two small matrices A and B are shown when replacing the parameter matrix of the fully connected layer, and the comparison of the number of parameters before and after is shown below

$$\frac{W_k}{W} = \frac{w_1 \times h_1 \times 2}{w_0 \times h_0}. \tag{2.5}$$

If $w_0 = w_1 \times w_1$ and $h_0 = h_1 \times h_1$, Eq (2.5) can be simplified to obtain

$$\frac{W_k}{W} = \frac{w_1 \times h_1 \times 2}{w_1 \times h_1 \times w_1 \times h_1} = \frac{2}{w_1 \times h_1}. \tag{2.6}$$

In the case, where w_1 and h_1 are large, applying the Kronecker product can greatly reduce the number of fully connected parameters of the parameterization.

3. The network structure

3.1. Encoding network

The encoding network uses the first four layers of the VGG16 network feature extraction module [25], and the size of the feature map is reduced by $1/2$ after each layer feature extraction. Each convolutional kernel size is 3×3. The first pooling layer of the encoding network is used for global feature extraction using the Kronecker product connectivity layer. The size of the first pooling layer is $H \times W \times C$, and after extending its feature space plane, two-dimensional features of size $(H \times W) \times C$ are obtained. The weight parameters constructed from the Kronecker product perform a "fully connected" operation on the first pooling layer, as shown in Figure 4.

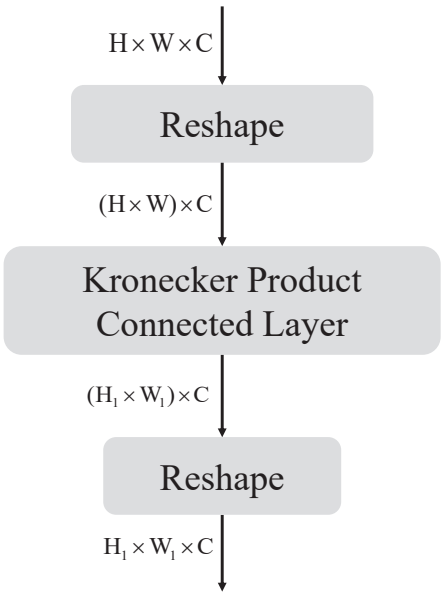


Figure 4. Kronecker product connection layer structure.

The 8-fold compressed feature maps are obtained after two Kronecker product-connected layers. The feature maps extracted by the Kronecker product and the layer-by-layer convolution are combined and filtered with two convolutional kernels of sizes 3×3 and 1×1 to obtain the output of the coded network, see Figure 5.

In addition, each convolutional layer contains a convolutional layer, a batch normalization layer, and a pooling layer, and the whole feature extraction structure is pruned to reduce the encoding network parameters and improve the network training speed.

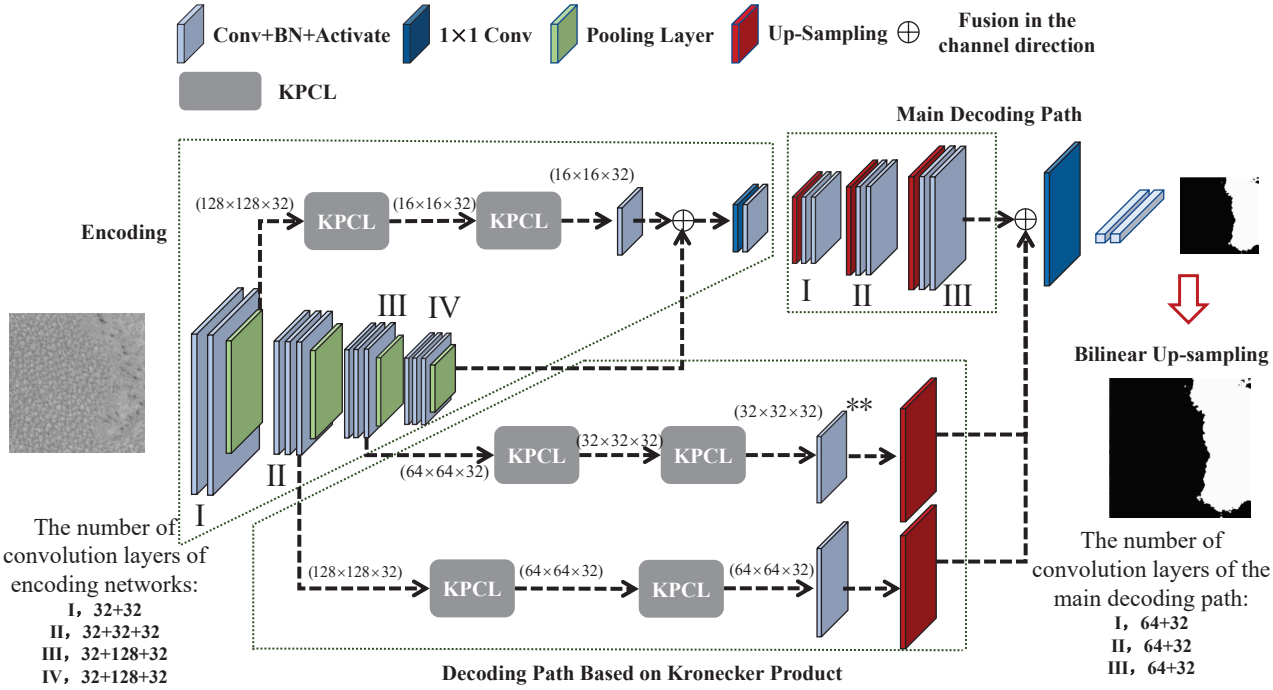


Figure 5. Semantic segmentation network structure based on Kronecker product multipath decoding.

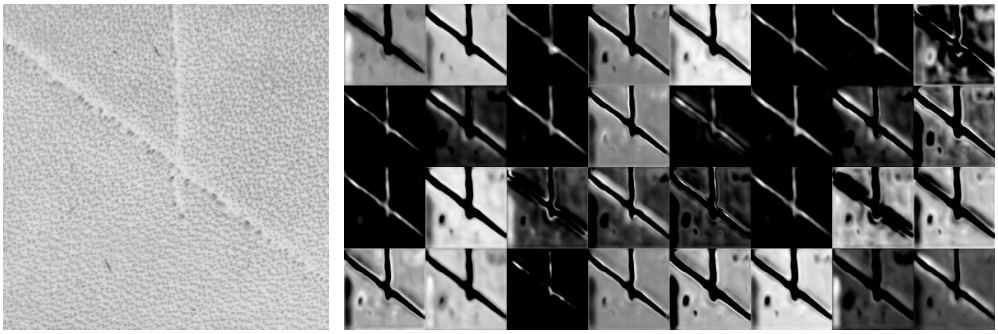
3.2. Decoding network

The decoding network uses three decoding paths, as shown in Figure 5. Of these, the first decoding path is a continuous 2-fold bilinear upsampling of the result of the encoding network, which yields a segmentation map that is 1/2 the size of the input. In addition, each small magnification of upsampling allows for better segmentation of small-sized defects. The other two decoding paths both rely on KPCL layers composed of the Kronecker product and target the feature maps before the second and third pooling layers in the encoding network. Both decoding paths use two KPCL layers, where the role of the first KPCL layer is to extract global semantic information from the feature map while performing 2-fold downsampling of the feature map, and the second KPCL layer and the subsequent 3×3 convolutional layers are responsible for optimizing the results of the first KPCL layer.

Then, the segmentation information extracted from the two decoding paths is directly amplified by bilinear upsampling and combined with the main decoding path to form three decoding paths. Finally, the results obtained from the three decoding paths are fused in the channel direction to obtain the output of KMDNet.

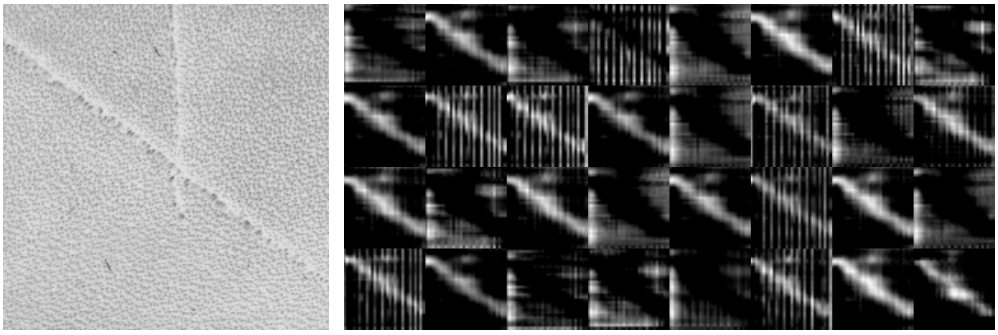
The three decoding paths scale the feature map to 1/2 the input size to reduce the parameters. The result of the network is still further scaled to the input size by additional bilinear interpolation. Figures 6 and 7 show the defect segmentation information after the Kronecker product decoding path and the main decoding path, of size $128 \times 128 \times 32$, which are obtained by stitching the defect segmentation information in the channel direction. Figure 6 shows the recognition effect of the main decoding path on the defective parts. It can be seen that most of the decoding results have a distinct black dot-like region in the lower-left corner, which is the same as the defective part, but the position in the origi-

nal defective image is a normal texture region. Figure 7 shows the results of the Kronecker product decoding path. As can be seen from the resulting image on the right, the result contains only the semantic information of the defect, and the lower-left corner does not contain additional defect information. Therefore, after combining the main decoding path and the Kronecker product decoding path, the wrong segmentation defect can be avoided in the lower-left corner of the segmentation result.



(a) The defect image (b) The output of the main decoding path

Figure 6. The result of the defect in the main decoding path.



(a) The defect image (b) The output of the Kronecker product decoding path

Figure 7. The result of the defect in the decoding path of the Kronecker product.

4. Experiment

This experiment uses Tensorflow and Keras as the deep learning development frameworks. The training platform is given below:

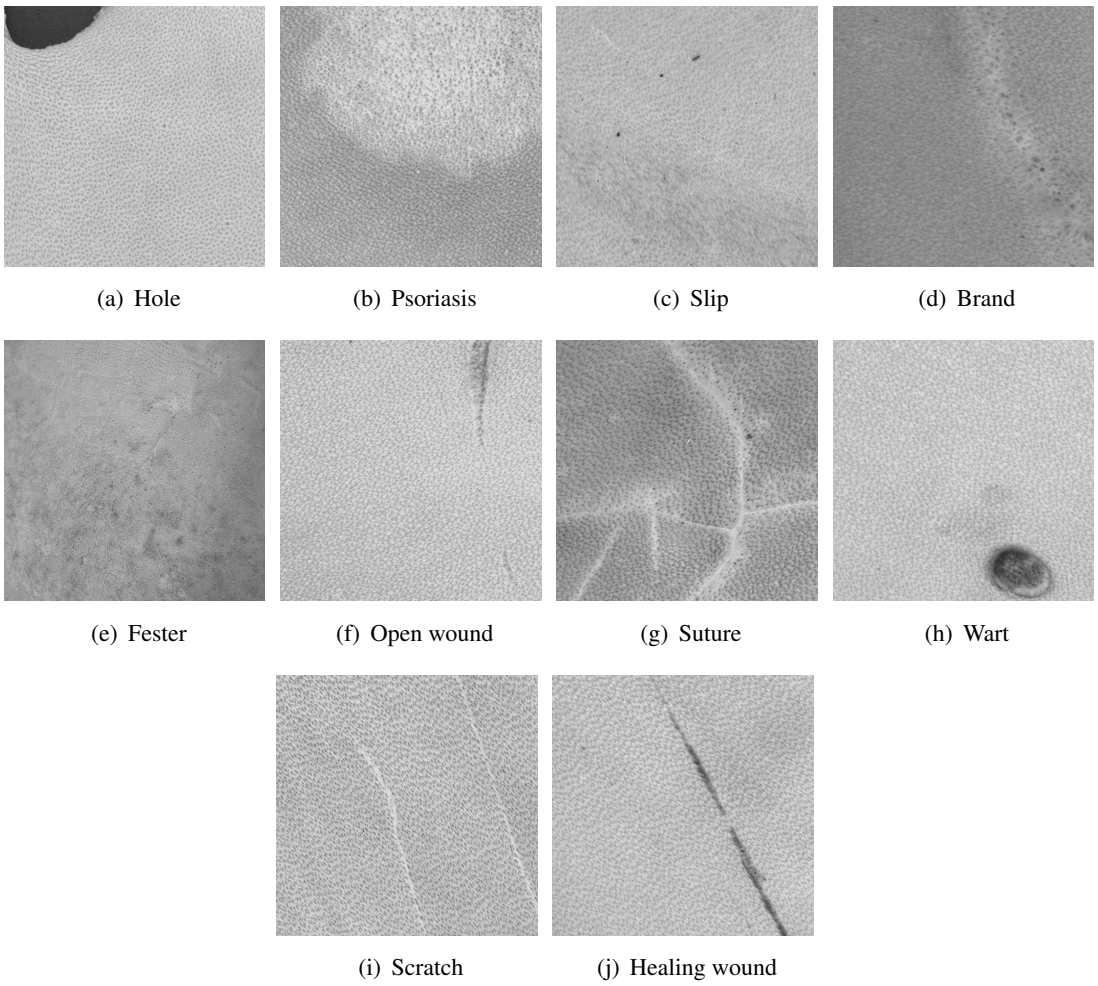
- CPU uses Intel (R) Weon (R) W-2102 CPU @2.9GHz (4-core).
- GPU uses Nvidia RTX TITAN, and the Video Memory is 24G.
- The system environment is Windows10.

and the testing platform is as follows:

- CPU uses AMD Ryzen5 1400 Quad-Core Processor @3.20GHz (4-core).
- The system environment is Windows10.

4.1. Dataset

The dataset for this experiment was collected by a related leather manufacturing plant in Jinjiang city using a high-precision industrial camera. The types of defects in the leather dataset include a hole, psoriasis, slip, brand, fester, open wound, suture, wart, scratch, and healing wound, as shown in Figure 8. The number of the wet leather datasets is 1944, and the ratio of defective samples to non-defective samples is 1603 : 524.



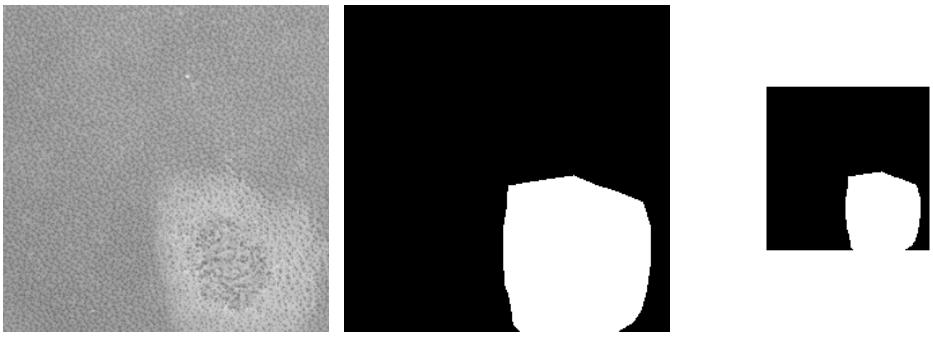
(a) Hole (b) Psoriasis (c) Slip (d) Brand

(e) Fester (f) Open wound (g) Suture (h) Wart

(i) Scratch (j) Healing wound

Figure 8. Wet blue leather samples with 10 types of defects.

To speed up the training time and reduce the testing time, the size of the original image is compressed to 256 dp $\times$ 256 dp and the defective regions of the image are labeled by the labeling procedure. Since the size of the network output is $1/2$ of the input, the image labels are also downsampled and the raw data labels are reduced to 128 dp $\times$ 128 dp to meet the needs of the model, see Figure 9.



(a) Original image (b) The same size label (c) Label after downsampling

Figure 9. Sample data.

4.2. Image preprocessing

Due to the large variety of defects in this training, some defects with inconspicuous features can greatly affect the effectiveness of the training. Therefore, appropriate image preprocessing is required to highlight the inconspicuous features of these defects while ensuring the details of the image texture. The preprocessing includes the following steps.

1) Gamma transform
   The gamma transform improves the detail of leather textures and imperfections by dividing the pixel value of the input image by 255, normalizing it to the interval [0, 1], and then calculating its gamma squared value, as follows:
   $$O_{(r,c)} = I^{\gamma}_{(r,c)} \times 255, \tag{4.1}$$
   where $I_{(r,c)}$ is the normalized pixel value. When $\gamma = 1$, the original pixel value is not affected; when $0 < \gamma < 1$, the more the pixel value, the higher the contrast of the picture; and $\gamma > 1$, the contrast of the picture is reduced.
2) Histogram equalization
   Histogram equalization can improve the contrast of an image. In this experiment, the grayscale histogram of the image needs to be counted after the gamma transformation of the image to improve the contrast.

4.3. Analysis of results

4.3.1. Evaluation criteria

In deep learning, the evaluation criteria for binary classification of machine learning models are based on the confusion matrix, see Table 1. In artificial intelligence, the confusion matrix is mainly used to compare the classification results with the actual measurements, and the accuracy of the classification results can be shown inside the confusion matrix. In Table 1, TP denotes the value where both the true category and the model prediction are 1; TN denotes the value where both the true category and the model prediction are 0; FP denotes the value where the true category is 0 and the model prediction is 1, and FN denotes the value where the true category is 1 and the model prediction is 0.

Table 1. Confusion matrix.

	Prediction: 1	Prediction: 0
True: 1	TP	FN
True: 0	FP	TN

The confusion matrix yields a number of model evaluation metrics. For example, Eq (4.2) shows the mean intersection over union ($mIoU$), it is a measure of the similarity between the predicted segmentation region and the true segmentation region in an image,

$$mIoU = \frac{1}{k+1}\sum_{i=0}^{k}\frac{TP}{FN+FP+TP}, \tag{4.2}$$

where k is the number of classification categories without background, and $k+1$ includes background categories. When $i = 0$, positive examples in the confusion matrix are the background, and negative examples are other target categories. When i is greater than 0, positive examples represent class i targets. In addition, the metric accuracy (ACC) measuring the proportion of correct classifications by the model is shown in Eq (4.3),

$$ACC = \frac{TP+TN}{TP+TN+FP+FN}. \tag{4.3}$$

Equation (4.4) displays a measure of the recall of the model in covering positive samples in the prediction,

$$Recall = \frac{TP}{TP+FN}, \tag{4.4}$$

Eq (4.5) gives the exponential $F1$ score,

$$F1 = 2\times\frac{Precision\times Recall}{Precision+Recall}, \quad \text{with} \quad Precision = \frac{TP}{TP+FP}, \tag{4.5}$$

which measures the accuracy of the dichotomous classification model.

The performance of the model was examined by using $mIoU$, ACC, $Recall$, and $F1$ score as comprehensive evaluation criteria, where the larger the values of $mIoU$ and ACC, the better the model.

4.3.2. Results analysis of the KPCL layer

To verify the role of the KPCL layer, the data results for the model without and with the KPCL layer are given in Table 2. It can be seen that the model with the KPCL layer removed decreases in all metrics, especially in $mIoU$ and $Recall$, which are severely attenuated. The metric $Recall$ represents whether the segmentation of leather defects in the image is complete, so the Kronecker product multi-decoding path improves the ability to segment leather defects completely without an increase in detection speed.

Table 2. Comparison results of KMDNet with and without multi-decoding path model.

Name	mIoU	ACC	Recall	F1
KMDNet	77.71%	93.43%	76.66%	79.52%
Model[1]	70.26%	92.22%	57.47%	73.19%

[1] It represents the KMDNet without a multi-decoding path based on the Kronecker product.

(a) Scratch (b) KMDNet (c) Model (d) Label

Figure 10. Graphics of scratch defect segmentation differences.

The KMDNet improved in all four metrics, especially the *mIoU* and *Recall* metrics, which improved by 7.45 and 19.19%, respectively. As can be seen in Figure 10, the KMDNet can segment the true defect areas on unimportant scratches, while the network without the KPCL layer cannot segment the same defects.

4.3.3. Comparison between different models

KMDNet is compared with models such as SegNet, U-Net, PSPNet, and DFANet, where U-Net [26] has been applied to leather defect detection and segmentation in recent years, SegNet [27] and PSPNet [28] are classical semantic segmentation models that have emerged recently, and DFANet [29] is a lightweight semantic segmentation network for road scenes that emerged in 2019. The segmentation data of each model under all defect types are shown in Table 3.

Table 3. Comparison results of KMDNet and other semantic segmentation network models.

Name	mIoU	ACC	Recall	F1	Time/Picture	Arguments
KMDNet	77.71%	93.43%	76.66%	79.52%	0.05 s	0.39 M
SegNet	67.25%	90.14%	66.00%	63.89%	0.157 s	29 M
U-Net	77.35%	93.79%	73.07%	77.53%	0.133 s	28 M
PSPNet	73.95%	93.04%	66.99%	73.33%	0.146 s	46 M
DFANet	68.38%	87.90%	67.02%	65.94%	0.113 s	2 M

"M" represents one million.

In order to use the network structure as the only variable, the same preprocessing process was performed on the data before the network training, and the sample weights and number of batches for the comparison model were kept consistent with KMDNet during training. Compared to DFANet, KMDNet has a 9.34% more *mIoU*, 5.53% more *ACC*, 9.64% more *Recall*, and 13.58% more $F1$ score, but the detection time is only half of the DFANet. Compared with U-Net, the *mIoU* of KMDNet increased by 0.36%, but the *ACC* decreased by 0.36%, and the two models were very close. However, for *Recall* and $F1$ score, KMDNet has more advantages and is significantly lower than U-Net in terms of detection speed and model parameters. This demonstrates that KMDNet has a shorter defect segmentation time and better recall compared to U-Net, which has been applied in the field of leather defect segmen-

tation. As shown in Figure 11, KMDNet has a higher overlap between the segmented defective part and the real defective part compared to U-Net. Moreover, KMDNet has many advantages over SegNet and PSPNet in all six metrics. Based on these metrics, KMDNet is the optimal solution for segmenting leather defects compared with SegNet, U-Net, DFAnet, and other networks.

(a) Scratch (b) KMDNet (c) UNet (d) Label

Figure 11. Graphics of scratch defect segmentation differences.

Figure 11 shows the different segmentation results of KMDNet and U-Net for the scratch defects; U-Net cannot completely segment the scratch defects, but KMDNet does a better job of segmenting the scratch defects. The difference between the two can also be reflected in the normal leather texture area.

(a) Defect-free leather image (b) KMDNet (c) UNet (d) Label

Figure 12. Difference graphs of defect-free leather image segmentation detection.

Figure 12 illustrates the different segmentation results of KMDNet and U-Net in the normal region. It can be seen that KMDNet accurately segmented the defect-free region, while U-Net showed segmentation errors.

Table 4 presents the segmentation results of KMDNet for different types of defects. As can be seen from it, defects such as the hole, psoriasis, brand, fester, and wart have the best segmentation results, with $F1$ scores above 80%. Defects such as open wounds, sutures, and healing wounds have the second-highest segmentation effect with an $F1$ score of over 73%, while slip and scratch have a poor segmentation effect with an $F1$ score of about 60%. This result indicates that there is a strong correlation between the feature saliency of defects and the segmentation effect. For defects with distinct features, the semantic segmentation network tends to learn better feature maps and thus achieve higher

segmentation results. However, for defects with unclear features, it is difficult for the network to learn better feature maps.

Table 4. Comparison of segmentation detection results for different defects.

Type	Hole	Psoriasis	Slip	Brand	Fester	Open wound	Suture	Wart	Scratch	Healing wound
F1 score	95.74%	87.08%	64.20%	85.95%	85.98%	73.09%	74.62%	81.61%	61.06%	75.55%

5. Conclusions

In this paper, a multi-decoding segmentation network using the Kronecker product, KMDNet, is proposed in order to identify defects in different shapes of wet blue leathers. The defect detection of wet blue leathers is more challenging than that of general rawhide leathers. This paper is an analytical study using a multi-decoding segmentation network with the Kronecker product. The following list is a summary of the major contributions of this paper.

1) Based on ensuring the network segmentation effect, our proposed KMDNet can reduce the number of parameters per convolution and the network output size; it can also reduce the number of network parameters and improve the detection speed.
2) KMDNet uses the Kronecker product to construct a new connectivity layer instead of a fully connected layer. The new connectivity layer will decode the feature map of the first layer of the network, extract global semantic information, and downsample it to fuse with the feature map of the regular feature extraction network in the channel direction.
3) The decoding network uses the Kronecker product connectivity layer to construct multiple decoding paths to decode the small-sized feature map to $1/2$ of the original image size and then obtains the final segmentation result by bilinear upsampling outside the network.
4) Trained with/through the collected wet blue leather defect data, our proposed KMDNet reveals a much better detection effect than the original segmentation network, and the $mIoU$ is improved by 7.45% after KMDNet is added to the KPCL layer.
5) KMDNet has advantages in segmentation effect compared with other networks such as U-Net, and at the same time, the number of network parameters is less and the detection speed is faster, which enables further practical application in the industry.

The model proposed in this paper is for the binary segmentation of wet blue leather defects. The coded network of KMDNet uses the KPCL layer for global semantic information extraction only for the first-level feature extraction unit. Our future work is to achieve the distinction of different types of defects while segmenting them, which requires enriching the network structure during the coding phase while keeping the network lightweight. The goal is to enable inspectors to effectively diagnose different defects faster. For example, convolutional layers are added to the encoding network to extract features that can represent the differences between different types of defects. For poorly segmented defects, the segmentation effect can be improved by increasing the amount of data for such defects. In addition, techniques such as near-infrared imaging can also be tried in the image preprocessing stage, allowing the segmentation network to extract richer defect information from the wet blue leather images. In the meantime, comparing KMDNet with the newly improved segmentation networks is also

part of our next work.

Acknowledgments

This research of F.R. is partially supported by Qing Lan Project of Jiangsu Province, NSFC (11601226), Project of Philosophy and Social Science Research in Colleges and Universities in Jiangsu Province (2021SJB0081), Postgraduate Research & Practice Innovation Program of Jiangsu Province (SJCX22_0406) and Postgraduate Education Reform Project of Nanjing Tech University (YJG2212). This research of J. H. is partially supported by Fujian Provincial Science and Technology Project Fund (2021T3060, 2021T3032) and Program of Quanzhou Science and Technology (2021C063L).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. C. Kwaka, J. A. Venturab, K. Tofang-Sazi, Automated defect inspection andclassification of leather fabric, *Intell. Data Anal.*, **5** (2001), 355–370. https://doi.org/10.3233/IDA-2001-5406
2. F. A. Faiz, A. Azhari, Tanned and synthetic leather classification based on images texture with convolutional neural network, *Knowl. Eng. Data Sci.*, **3** (2020), 77–88. http://dx.doi.org/10.17977/um018v3i22020p77-88
3. Y. T. Lee, C. Yeh, Automatic recognition and defect compensation for calf leather, *Int. J. Inf. Technol. Manage.*, **19** (2020), 93–117. https://doi.org/10.1504/IJITM.2020.106211
4. M. Jawahar, L. J. Anbarasi, S. G. Jasmine, M. Narendra, R. Venba, V. Karthik, A machine learning-based multi-feature extraction method for leather defect classification, in *Inventive Computation and Information Technologies*, **173** (2021), 189–202. https://doi.org/10.1007/978-981-33-4305-4_15
5. Y. S. Gan, S. T. Liong, S. Y. Wang, C. T. Cheng, An improved automatic defect identification system on natural leather via generative adversarial network, *Int. J. Computer Integr. Manuf.*, **2022** (2022), 1–17. https://doi.org/10.1080/0951192X.2022.2048421
6. Y. S. Gan, W. C. Yau, S. T. Liong, C. C. Che, Automated classification system for tick-bite defect on leather, *Math. Probl. Eng.*, **2022** (2022), 5549879. https://doi.org/10.1155/2022/5549879
7. T. Adao, D. Gonzalez, Y. C. Castilla, J. Perez, S. Shahrabadi, N. Sousa, et al., Using deep learning to detect the presence/absence of defectson leather: on the way to build an industry-driven approach, *J. Phys. Conf. Ser.*, **2224** (2022), 012009. https://doi.org/10.1088/1742-6596/2224/1/012009
8. F. López, J. M. Prats, A. Ferrer, J. M. Valiente, Defect detection in random colour textures using the MIA T^2 defect maps, in *ICIAR 2006: Image Analysis and Recognition*, (2006), 752–763. https://doi.org/10.1007/11867661_68

9. R. Viana, R. B. Rodrigues, M. A. Alvarez, H. Pistori, SVM with stochastic parameter selection for bovine leather defect classification, in *PSIVT 2007: Advances in Image and Video Technology*, (2007), 600–612. https://doi.org/10.1007/978-3-540-77129-6_52
10. H. Q. Bong, Q. B. Truong, H. C. Nguyen, M. T. Nguyen, Vision-based inspection system for leather surface defect detection and classification, in *2018 5th NAFOSTED Conference on Information and Computer Science (NICS)*, (2018), 300–304. https://doi.org/10.1109/NICS.2018.8606836
11. C. Kwak, J. A. Ventura, K. Tofang-Sazi, A neural network approach for defect identification and classification on leather fabric, *J. Intell. Manuf.*, **11** (2000), 485–499. https://doi.org/10.1023/A:1008974314490
12. A. Varghese, S. Jain, A. A. Prince, M. Jawahar, Digital microscopic image sensing and processing for leather species identification, *IEEE Sens. J.*, **20** (2020), 10045–10056. https://doi.org/10.1109/JSEN.2020.2991881
13. D. M. Tsai, T. Y. Huang, Automated surface inspection for statistical textures, *Image Vision Comput.*, **21** (2003), 307–323. https://doi.org/10.1016/S0262-8856(03)00007-6
14. J. W. Kwon, Y. Y. Choo, H. H. Choi, J. M. Cho, G. S. KiI, Development of leather quality discrimination system by texture analysis, in *2004 IEEE Region 10 Conference TENCON 2004*, **1** (2004), 327–330. https://doi.org/10.1109/TENCON.2004.1414423
15. K. Krastev, L. Georgieva, Identification of leather surface defects using fuzzy logic, in *2005 International Conference on Computer Systems and Technologies*, (2005), IIIA.12-1–IIIA.12-6. Available from: http://ecet.ecs.uni-ruse.bg/cst05/Docs/cp/SIII/IIIA.12.pdf.
16. D. H. Fan, L. Ding, J. H. Deng, Automatic detection and localization of surface defects for whole piece of ultrahigh-definition leather images, in *2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*, (2019), 229–232. https://doi.org/10.1109/CCOMS.2019.8821662
17. S. T. Liong, D. Zheng, Y. C. Huang, Y. S. Gan, Leather defect classification and segmentation using deep learning architecture, *Int. J. Computer Integr. Manuf.*, **33** (2020), 1105–1117. https://doi.org/10.1080/0951192X.2020.1795928
18. J. Wang, G. Yi, S. Zhang, Y. Wang, An unsupervised generative adversarial network-based method for defect inspection of texture surfaces, *Appl. Sci.*, **11** (2020), 283. https://doi.org/10.3390/app11010283
19. S. Y. Chen, Y. C. Cheng, W. L. Yang, M. Y. Wang, Surface defect detection of wet-blue leather using hyperspectral imaging, *IEEE Access*, **9** (2021), 127685–127702. https://doi.org/10.1109/ACCESS.2021.3112133
20. Y. Shen, T. Xiao, S. Yi, D. Chen, X. Wang, H. Li, Person re-identification with deep kronecker-product matching and group-shuffling random walk, *IEEE Trans. Pattern Anal. Mach. Intell.*, **43** (2019), 1649–1665. https://doi.org/10.1109/TPAMI.2019.2954313
21. Z. J. Xiao, X. D. Yang, X. Wei, X. L. Tang, Improved lightweight network in image recognition, *J. Front. Comput. Sci. Technol.*, **15** (2021), 743–753. https://doi.org/10.3778/j.issn.1673-9418.2004057

22. H. V. Henderson, F. Pukelsheim, S. R. Searle, On the history of the Kronecker product, *Linear Multilinear Algebra*, **14** (1983), 113–120. https://doi.org/10.1080/03081088308817548

23. T. Wu, S. Tang, R. Zhang, J. Cao, J. Li, Tree-structured kronecker convolutional network for semantic segmentation, in *2019 IEEE International Conference on Multimedia and Expo (ICME)*, (2019), 940–945. https://doi.org/10.1109/ICME.2019.00166

24. F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, preprint, arXiv:1511.07122.

25. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, preprint, arXiv:1409.1556.

26. O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, (2015), 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

27. V. Badrinarayanan, A. Kendall, R. Cipolla, Segnet: A deep convolutional encoder-decoder architecture for image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.*, **39** (2017), 2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615

28. H. S. Zhao, J. P. Shi, X. J. Qi, X. G. Wang, J. Y. Jia, Pyramid scene parsing network, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2017), 2881–2890. https://doi.org/10.1109/CVPR.2017.660

29. H. C. Li, P. F. Xiong, H. Q. Fan, J. Sun, Dfanet: Deep feature aggregation for real-time semantic segmentation, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (2019), 9522–9531.

AIMS Press