applied sciences

MDPI

Article

Deep Feature Retention Module Network for Texture Classification

Sung-Hwan Park, Sung-Yoon Ahn and Sang-Woong Lee *

Pattern Recognition and Machine Learning Lab, Department of AI·Software, Gachon University, Seongnam 13120, Republic of Korea; eodlf311@gachon.ac.kr (S.-H.P.); sungyoonahn@gachon.ac.kr (S.-Y.A.)
* Correspondence: slee@gachon.ac.kr

Abstract: Texture describes the unique features of an image. Therefore, texture classification is a crucial task in computer vision. Various CNN-based deep learning methods have been developed to classify textures. During training, the deep-learning model undergoes an end-to-end procedure of learning features from low to high levels. Most CNN architectures depend on high-level features for the final classification. Hence, other low- and mid-level information was not prioritized for the final classification. However, in the case of texture classification, it is essential to determine detailed feature information within the pattern to classify textures as they have diversity and irregularity in images within the same class. Therefore, the feature information at the low- and mid-levels can also provide meaningful information to distinguish the classes. In this study, we introduce a CNN model with a feature retention module (FRM) to utilize features from numerous levels. FRM maintains the texture information extracted at each level and extracts feature information through filters of various sizes. We used three texture datasets to evaluate the proposed model combined with the FRM. The experimental results showed that learning using different levels of features together assists in improving learning performance more than learning using high-level features.

Keywords: texture classification; feature information; numerous level; FRM

Citation: Park, S.-H.; Ahn, S.-Y.; Lee, S.-W. Deep Feature Retention Module Network for Texture Classification. *Appl. Sci.* **2024**, *14*, 4011. https://doi.org/10.3390/app14104011

Academic Editors: Hyeonjoon Moon and Lien Minh Dang

Received: 22 March 2024
Revised: 1 May 2024
Accepted: 7 May 2024
Published: 9 May 2024

1. Introduction

Texture is an important image characteristic in various fields [1,2]. In particular, texture provides meaningful cues for recognizing features in images [3,4]. To examine these characteristics, general object-based and texture-based datasets are compared with each other as follows: object-based datasets, such as ImageNet [5], CIFAR100 [6], and CUB-200-2011 [7], have been used to train the general model. Figure 1a shows examples of images from CUB-200-2011, in which images of 200 different objects were collected. For example, the first and third rows can be distinguished because Bobolink and Florida Jay have different characteristics. Similarly, during the training process, an object-based dataset can be used to determine the distinguishing characteristics between various objects and then classify those objects. However, in contrast to object-based datasets, texture-based datasets consist of unique image patterns. The classes in these datasets contain images that do not belong to the same objects. As shown in Figure 1b, which shows examples of images in the describable textures dataset (DTD) [8], most images in each class are made of the microparts of the objects, showing their material pattern.

Because CNN-based models exhibit high performance in the ImageNet competition, various studies on deep learning have been conducted in computer vision. The typical deep learning models include AlexNet [9], VGG [10], ResNet [11], DenseNet [12], and EfficientNet [13]. These models commonly consist of a convolutional neural network (CNN) to learn data characteristics while maintaining the input data characteristics and preventing information loss during learning [14,15]. During the learning process, insufficient data can affect the accuracy of the model. One solution to this problem is transfer learning [16]. Using transfer learning, models can be pretrained on publicly available large datasets, such as ImageNet, which can be further trained on the target dataset with fewer data.

Additionally, fine-tuning techniques can be used to adjust the structure of the pretrained model to suit the current purpose. Because texture datasets lack the number of images and classes, a model structure should be constructed to identify the features of the texture dataset by fine-tuning. As CNN models suffer from the problem of vanishing gradients with increasing layer depth, it is difficult to expect high-performance results when using a typical model structure. Thus, CNN-based models that classify texture datasets commonly use structures that retain the characteristics of transfer learning models. For example, ref. [17] presented a FASON model with a module that combined features learned from multiple-level information. To maintain the detailed characteristics of the texture, ref. [18] presented a learnable encoding module (LEM) that combined the low-to-high-level features of a CNN. The models proposed in [17,18] commonly applied additional module types that preserved texture information using pretrained models to preserve different layer features and minimize loss. These additional features result in an improvement in the classification performance between classes.

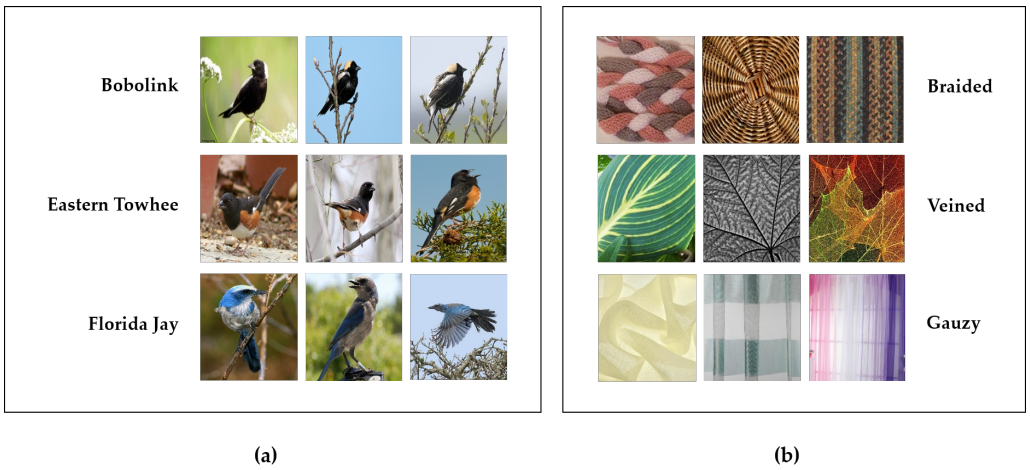

Figure 1. (**a**,**b**) refer to examples of object- and texture-based datasets, respectively.

In contrast to object classification, texture classification is a method for identifying unique patterns in materials in images. Various experiments have shown that it is advantageous to extract the comprehensive characteristics of an image and use them together to determine common regularities in the diversity of texture patterns. In addition, it has been demonstrated that extracting and using detailed feature information from the output of various levels results in performance improvement [3,18]. Therefore, in this study, we propose a new Deep Feature Retention Module Network to classify texture datasets. First, inspired by the structure proposed in [18], various level characteristics were utilized. The aim was to determine the detailed features of the texture and improve the classification performance between classes by maintaining texture information from low to high levels. Second, a transfer learning method was used for training on small-scale texture datasets. Third, a module using convolutional filters of various sizes was proposed to extract diverse features using a pretrained model and to determine detailed features within the textures. This module was inspired by GoogleNet [19].

The remainder of this paper is organized as follows: In Section 2, we review the related studies. Section 3 provides a detailed description of the proposed model and module. Section 4 presents the experiments and results for the texture datasets. Additional experiments on the modified module are described in Section 5. Finally, Section 6 presents the conclusions.

2. Related Works

2.1. Model for Texture Classification

The texture shows the characteristics of the images. It is an essential element for recognizing and classifying images in computer vision and is considered a significant problem to be solved because it can be witnessed all over. Until CNN gained prominence in image classification in 2012, local-based manual techniques, such as bag of words (BoW) [20–22], scale invariant feature transform (SIFT) [23], and speed-up robust features (SURF) [24], were predominant. CNN-based methods were first proposed in 2012. To overcome the limitations of small datasets, transfer learning, which is a technique that uses a model learned with large datasets, was used. In addition, fine-tuning, which modified the structure of the model to suit the purpose of learning and adjusted the weights of the transfer learning model, was applied. Using this technique, model structures for identifying and maintaining fine features between textures were presented. Both ScatNet [25] and PCANet [26] proposed similar structures that improved model performance by connecting input images with feature vectors extracted from each convolutional layer using feature pooling techniques. Moreover, it is difficult to distinguish the unique characteristics of texture from minute differences between classes. Therefore, similar to global features, local information contains essential information that can distinguish subtle differences between classes. Ref. [18] presents MuLTER with a multilevel pooling structure that uses all low-to-high-level features to maintain texture details and spatial information.

2.2. Multiple Structure to Obtain Features

The CNN structure extracts the feature information of classes from the input images using convolutional layers during the learning process. However, owing to the structural nature of CNN, the final classification is affected by high-level feature information. However, information other than that at high levels is meaningful for texture classification. Therefore, various concepts have been presented to utilize the features of each level. Ref. [18] suggested a method for connecting low-to high-level information to the proposed module and using it for learning to maintain and utilize texture feature information and spatial information. In [3], feature maps were extracted for each module using the inception-v3 structure, and the characteristics of all levels were utilized. They also presented a structure that combined principle component analysis (PCA) [27] and linear discriminant analysis (LDA) techniques to minimize the influence of less relevant factors.

2.3. Feature Module for CNN

In CNN models, depth and width are factors that improve performance; however, the amount of computation and the number of parameters increases, resulting in the problems of learning being time-consuming, a loss of information on input values, or gradient vanishing on depth. Various structures have been proposed to address these issues. In [11], the proposed residual block solved the gradient vanishing problem by adding a skip connection, in which an input value was added to the output value of the existing network structure. In addition, a structure using a residual block in a VGG-based structure was proposed to learn features without losing information from previous layers, even if the network was deepened. DenseBlock [12] used a concatenation method rather than adding the feature map of the last layer to the input of the next layer. Consequently, reusing the features with a structure that connected the layers of the entire model mitigated the effects of the vanishing gradient problem. GoogleNet proposed an inception module to address issues by constructing a deep neural network and extracting various features. In the inception module, the input values are calculated using multiple-sized convolutional filters (1×1, 3×3, and 5×5) and 3×3 max pooling in parallel and then connected, comprehensively combined, and exported as output values. Furthermore, by connecting these modules, they constructed a network with deep layers and used an auxiliary loss layer in the middle of the model to solve the gradient vanishing problem caused by deeper layers.

3. Proposed Method

3.1. Model Architecture

The structure of DenseNet is shown in Figure 2a. DenseNet has a DenseBlock comprising several layers. The input value of the previous layer was closely linked to that of the next layer using DenseBlock, allowing the feature information of the previous layer to be reused. DenseNet is also a model developed to preserve different levels of information compared to ResNet, which uses a skip connection technique to solve the gradient loss problem as the network deepens. Because of these structural characteristics, DenseNet is believed to be more helpful for extracting and utilizing detailed feature information about textures than general models such as VGG. The structure shown in Figure 2b is proposed based on the characteristics of DenseNet. The proposed model has a multiple-feature structure that can consider all features, from low to high levels, in the learning process. The output information generated for each DenseBlock is extracted using various features through each feature retention module (FRM). The generated output features are concatenated and used for the final classification. The values extracted and combined through the structure of this model contain features identified at each level, and preserved information. In addition, because these values are used through the structure proposed during learning, they can help one classify patterns of texture datasets with regularity and irregularity.

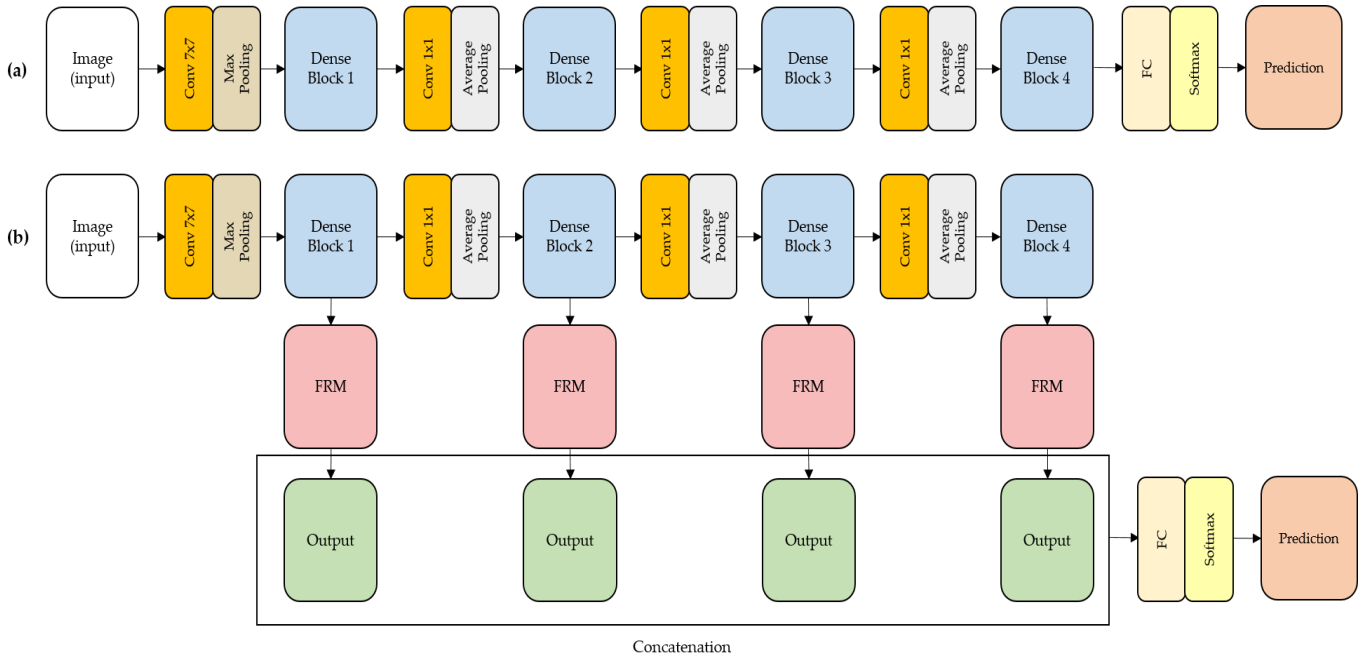

Figure 2. (**a**) DenseNet structure. (**b**) shows the proposed model structure based on DenseNet as the backbone.

The expected effects of the proposed model structure based on DenseNet are as follows:

- DenseNet has a structural characteristic that uses information from the input layer connected to the output layer in the feed-forward process. Therefore, it can consider the advantages of features and continually learn all feature information from low to high levels.
- FRM was used for each DenseBlock to minimize the loss of information as the layers of the backbone model deepened. This is expected to have an important influence on the use of the detailed features of textures for learning.
- We aimed to use various detailed features of the texture dataset for learning. Therefore, the structural characteristics of the model proposed in [18] were referred to. By combining the FRM with the backbone, it is expected that the outputs through the

FRM from low to high levels are aggregated and can help distinguish classes with high probability.

3.2. Feature Retention Module

The structure of the proposed FRM is shown in Figure 3. The output value of each block was divided into local and global parts to extract the features. In the case of the local part, features were extracted and utilized through convolutions of various sizes. We used these convolutions based on the structural characteristics of the inception module presented in GoogleNet. Figure 4 shows two types of inception module structures. Figure 4a shows the inception module of the basic structure, and Figure 4b shows the dimension reduction. We refer to the structure presented in Figure 4b to obtain the calculation reduction effect and extract the characteristics of various convolutions. Three different filters were used as convolutional filters for extracting features 1×1, 3×3, and 5×5. In contrast to the max pooling of 3×3 in the inception module in Figure 4b, the max pooling of 1×1 was used to obtain more detailed information. Furthermore, each convolutional part had a convolution structure, batch normalization, and a ReLU. In the case of the global part, conv 1×1 was used as the value from DenseBlock for the effect of channel reduction, similar to the local part. A feature map was generated using average pooling to obtain the global characteristics of the output value of the block. Finally, the two values from local and global were concatenated to create a new output. The new output contained numerous meaningful features extracted from an image. Therefore, it is expected that extracting feature information through FRM for each output of DenseBlock can improve the model classification performance.

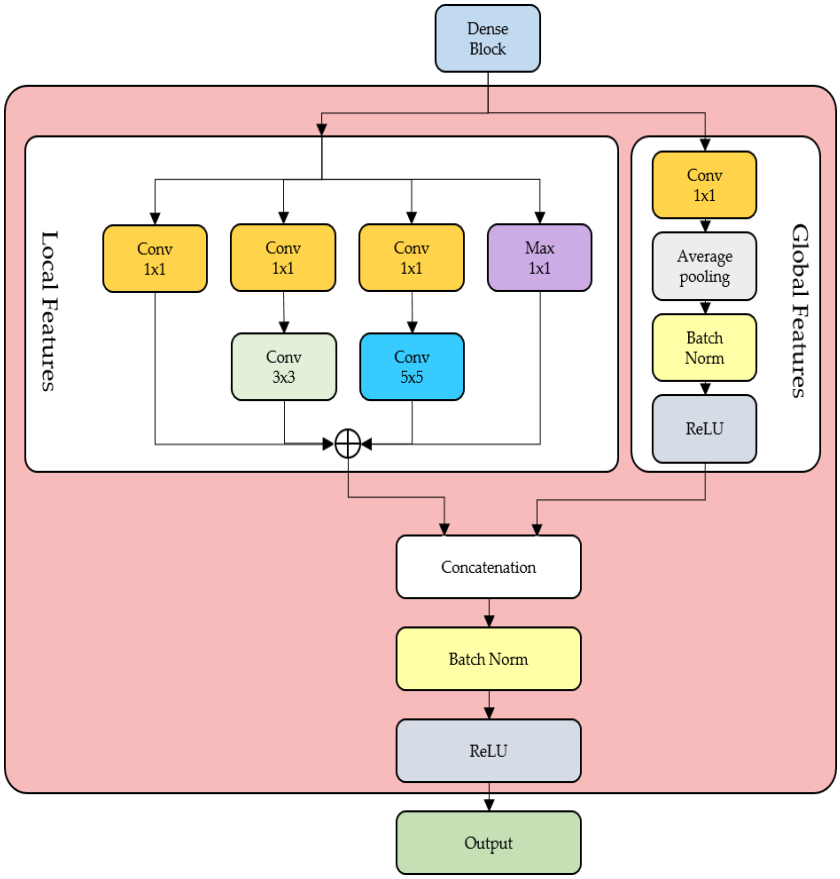

Figure 3. The architecture of proposed feature retention module (FRM).

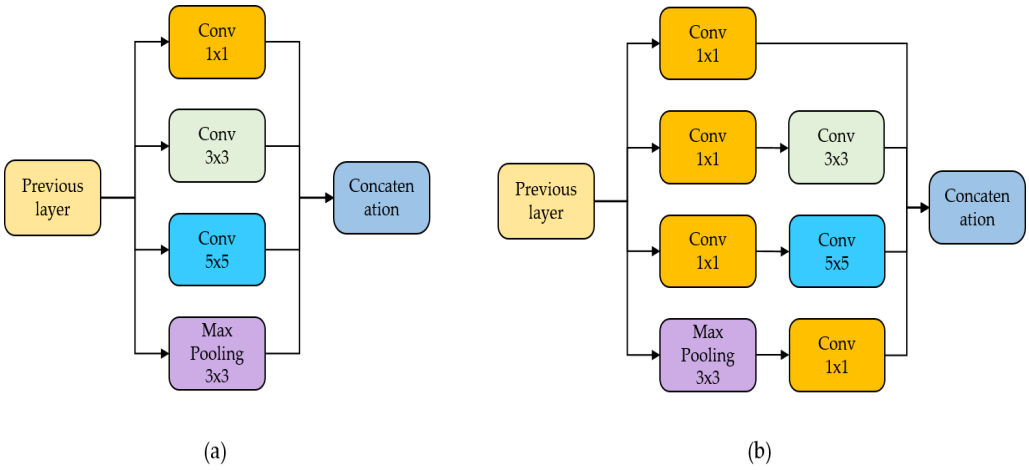

Figure 4. (**a**) shows the basic structure of the inception module, and (**b**) shows the type of inception module used for dimension reduction.

4. Experiment

4.1. Dataset

In this experiment, the DTD [8], Flickr Material Database (FMD) [28], and KTH-TIPS2b [29], which are publicly available for texture classification, were used for training and testing. The DTD has 47 classes, consisting of 120 images per class. It is a dataset composed of images with perceptual properties of textures, such as grids, spider webs, and spirals. The FMD is composed of 10 classes, consisting of 100 images per class. It consists of images captured from various materials, such as fibers, glass, and water, from proximity or general distance. The KTH-TIPS2b has 11 classes, with 432 images per class, and is a dataset consisting of images captured closely with various poses, lighting, and scales. Classes such as wood, wool, and cotton are included in the dataset. The characteristics of each dataset are as follows: The FMD is a dataset composed of images of cropped parts of objects. Irregularities were observed in the images, even within the same class. For example, the same class of plastics in Figure 5b had different shapes in each image. The KTH-TIPS2b is a dataset consisting of images in which all classes are patterns of objects within a close range. Hence, regularity exists within the same class. Finally, the DTD is a mixture of the FMD and the KTH-TIPS2b characteristics. Because each class consisted of images captured from a close range or cropped images to focus on the material, regularity or irregularity exists within the same class.

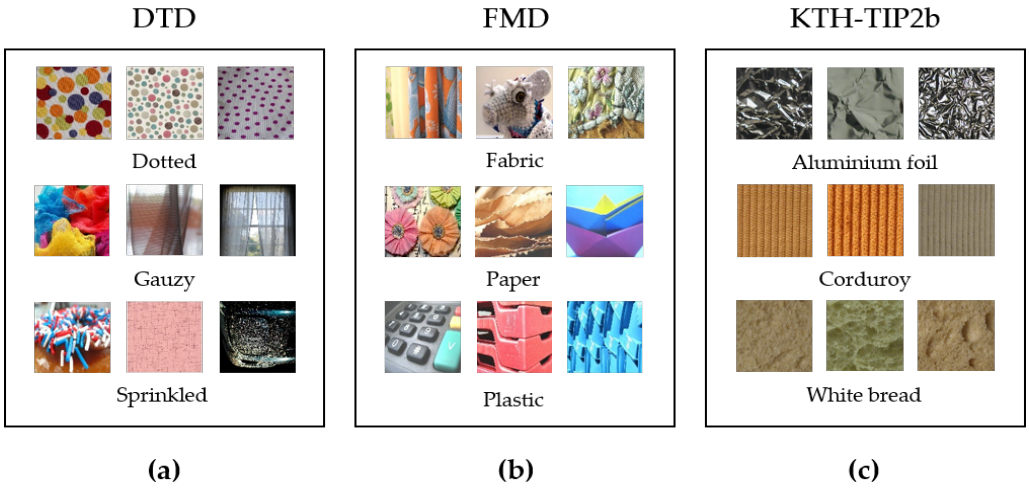

Figure 5. (**a**–**c**) are example images for the texture dataset, respectively.

4.2. Implementation Setting

To evaluate the proposed model using the three datasets, a computer with RTX2080ti (NVIDIA, Santa Clara, CA, USA) was used. DenseNet161 [12] was used as the backbone, and the proposed model was implemented using the Pytorch framework. The Adam optimizer was used with a learning rate of 0.0001. The batch size and number of epochs were set to 50 and 500, respectively. RandomSizecrop and RandomHorizontalFlip were used for data augmentation. The input images were applied differently based on the dataset, considering the image size in the dataset. Because DTD and FMD consisted of images with a size of 300 × 300 or more, the input image size was set to 224 × 224. However, because KTH-TIPS2 is composed of images with a size of 200 × 200 or less, the input image size was set to 64 × 64, considering the smallest image size. For the experiments, each dataset was split into seven for training and three for testing. We adopted accuracy as the main evaluation metric for the experiments.

4.3. Experiment Method and Result

The output values of each DenseBlock were concatenated into four combinations to analyze the impact of multiple levels in various settings, as shown in Figure 6. As shown in Figure 6a, the output values of each DenseBlock are concatenated, and the feature information of all levels is used for learning. Subsequently, from Figure 6b–d, the structure of the model is constructed and learned by sequentially excluding the DenseBlock. Figure 6d shows a structure in which only high-level features were used. Additionally, each method commonly included the last DenseBlock, which outputted high-level features that significantly influenced classification performance. The learning results of the model learned in the four combinations are listed in Table 1, and the four combinations are sequentially listed in Table 1 as Blocks 14, 24, 34, and 4. The results from Blocks 14 to 4 showed that the structure of Block 14, used by combining all levels of output values, was more effective for learning than Block 4, which used only high-level output values. In other words, when using the proposed structure with the FRM, it was confirmed that the datasets with regularity and irregularity in the image were effectively classified. In addition, the proposed method on the KTH-TIPS2b dataset with regularity within classes was effective in terms of performance.

Table 1. Evaluation results on texture datasets. Bold denotes best accuracy.

Method	Backbone	DTD	FMD	KTH-TIP2b
BP-CNN [30]	VGG19	69.6	77.8	75.1
FV-CNN [31]	VGG19	72.3	79.8	75.4
LFV [32]	VGG19	73.8	82.1	82.6
FASON [17]	VGG19	72.9	82.1	76.5
DeepTEN [33]	ResNet50	69.6	80.2	82.0
LSCNet [34]	VGG16	71.1	82.4	76.9
CLASSNet [4]	ResNet18	71.5	82.5	85.4
CLASSNet [4]	ResNet50	74.0	**86.2**	87.7
Block 14 (Ours)	DenseNet161	**74.53**	82.76	**92.97**
Block 24 (Ours)	DenseNet161	74.52	82.07	91.55
Block 34 (Ours)	DenseNet161	74.35	81.72	91.83
Block 4 (Ours)	DenseNet161	74.35	81.72	91.76

The analysis of the experimental results for each dataset is as follows: In the DTD results, the proposed methods showed higher accuracy than the results of the previously proposed methods. In particular, in the case of Block 14, using all the feature information from low to high levels, the highest result was 74.53%. This is a +4.93% increase compared to BP-CNN (69.6%) and a +0.18% increase compared to Block 4, which used only high-level features. It was identified that images with irregular shapes between classes were efficiently classified when different features were used together rather than using only high levels.

The results of the FMD showed that the accuracy of Block 14 using different levels of features was higher than that of the previous cases, except for CLASSNet. In addition, among the proposed methods, Block 14 showed an increase of +1.04% compared to Block 4 (81.72%). Similar to the results of DTD, the combination of different levels of features when classifying images with irregular forms is effective at extracting detailed information within the image. In the dataset KTH-TIPS2b, the accuracy of Block 14 was higher than the others. Compared with the results of Block 4, Block 14 showed better accuracy. The results proved that, similar to other datasets, it is advantageous to use various levels rather than only high levels to analyze datasets. However, exceptionally, the result of Block 24 showed some performance reduction. Block 24 has a structure that does not additionally combine features for DenseBlock 1. In Table 1, the performance of Block 24 is slightly lower, but Block 14 combined with all blocks improves performance again. If there are many holes in the bread, such as in the case of white bread in Figure 5c, more detailed information is needed to distinguish bread. In other words, when the low level is used, it is determined that information of the DenseBlock1 is also necessary.

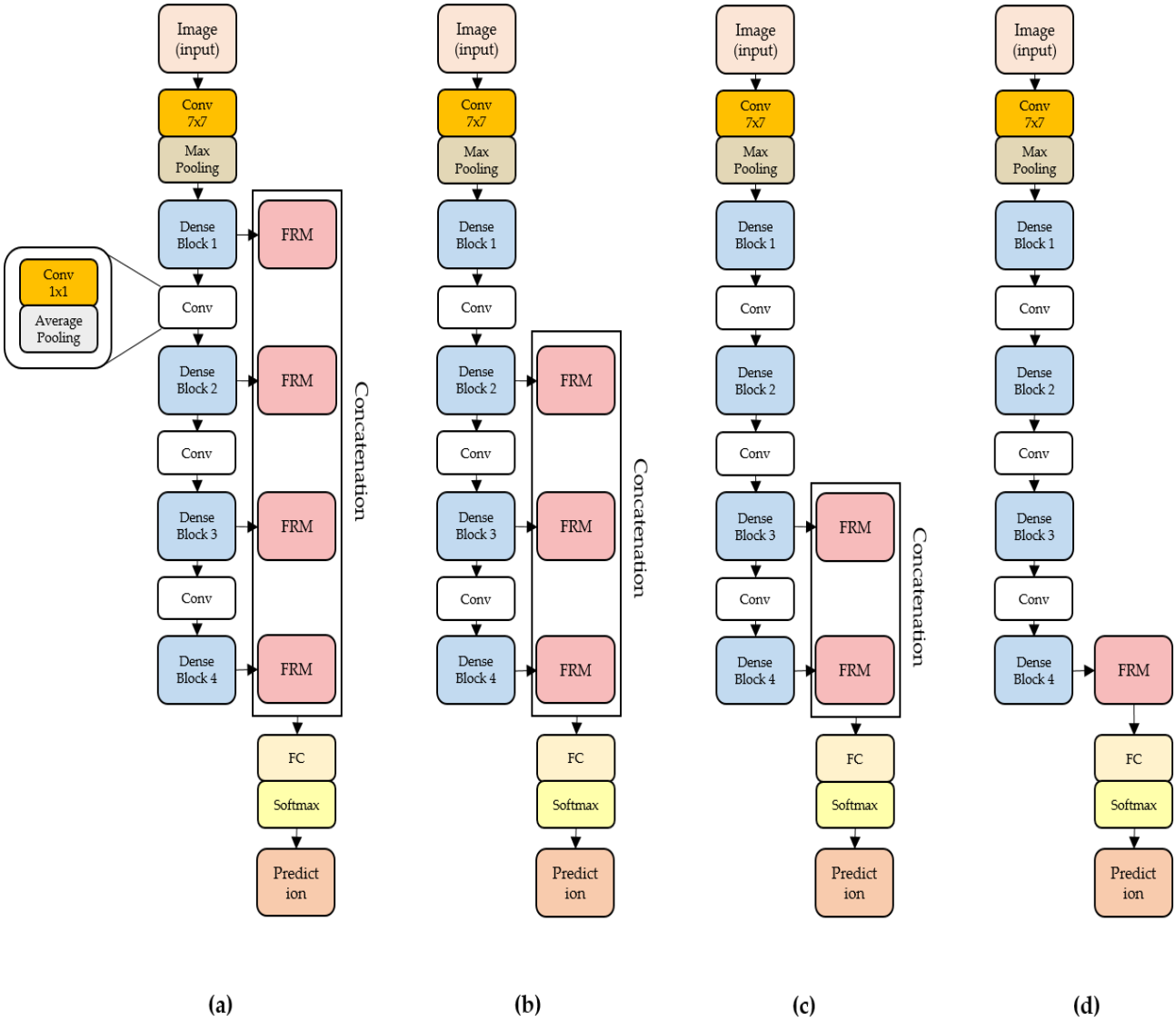

Figure 6. (**a**–**d**) are combination examples of the proposed model for evaluating datasets. (**a**–**d**) are referred to as Block 14, Block 24, Block 34, and Block 4, respectively.

A comprehensive review of the experimental results is as follows: The dataset used in the experiment consisted of images cropped to specific patterns for each class. The images exhibited regularities and irregularities within the same class. For example, Figure 5b refers to the FMD dataset, and the images have different shapes for each class. That is, since each image has irregular features, it means that the similarity of feature information of the same class is low. On the other hand, the KTH-TIP2b dataset in Figure 5c shows that each class has similar images. It also shows that the similarity of texture feature information between images is high because each image has regular features. And we used a structural model that utilized features at multiple levels to maximize the features of the pattern in the image. Because of the structural features combined with FRM, it is of use to extract detailed feature information for irregular and regular images. In particular, feature information on regular images from KTH-TIP2b is useful for improving performance as it has high similarity and thus can be used to distinguish between classes. Thus, the experiments demonstrated that using all the characteristics of low to high levels together was effective for classifying datasets. In other words, the experimental results showed that the proposed model was effective at classifying meaningful patterns in the texture dataset.

4.4. Dataset Analysis

We analyzed a confusion matrix for the FMD based on the results in Section 4.3. The results of the confusion matrix are shown in Figure 7. Figure 7 shows the results for Blocks 14, 24, 34, and 4 in the order of a to d. The overall true positive (TP), which predicts that the actual true value is true, is organized. However, there are classes that correspond to false negatives (FN) that falsely predict the actual true values. Among them, we selected the images of classes not commonly classified in Figure 7 from a to d. Figure 8 shows examples of the four classes corresponding to the FN. The actual class in Figure 8a was glass; however, it was incorrectly predicted to be water. Figure 8b,c show metal and stone, respectively; however, they were predicted to be wood, and Figure 8d was predicted to be metal even though the actual class was wood. Figure 9 shows examples of images from the FMD dataset. A comparative analysis of the images in each class was performed as follows: Compared to the images of water in Figure 9a, the images of glass in Figure 8a have similarities in terms of shape and reflection. Figure 8b,c represent wood. They have characteristics very similar to those of wood in terms of pattern, reflection shape, and texture (Figure 9c). In addition, the images in Figure 8d are classified as metal. They have related features such as patterns and textures, as shown in Figures 8b and 9b. In conclusion, the FN results of the actual and predicted classes were fairly similar. Some images have similarities that are difficult to distinguish with the naked eye.

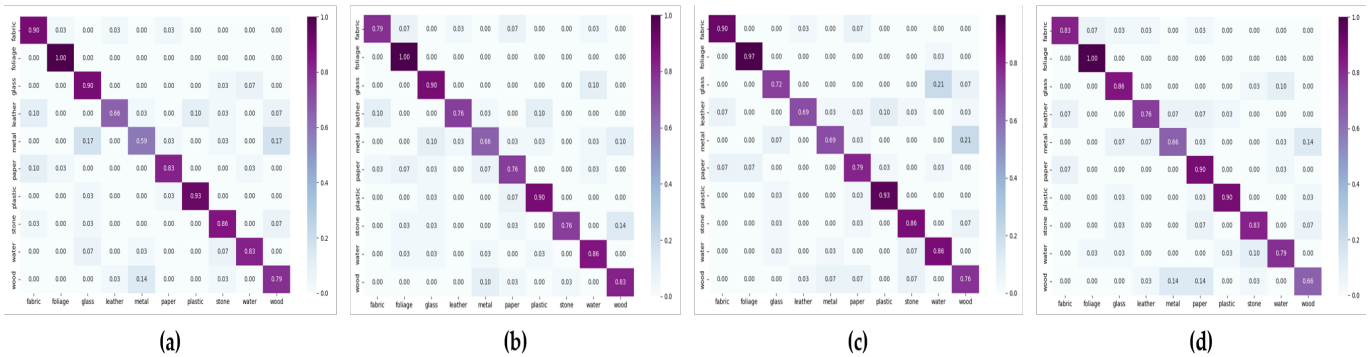

Figure 7. (**a**–**d**) show the confusion matrix of Blocks 14, 24, 34, and 4 on FMD, respectively. In the confusion matrix, the y-axis and x-axis denote actual class and predicted class, respectively.

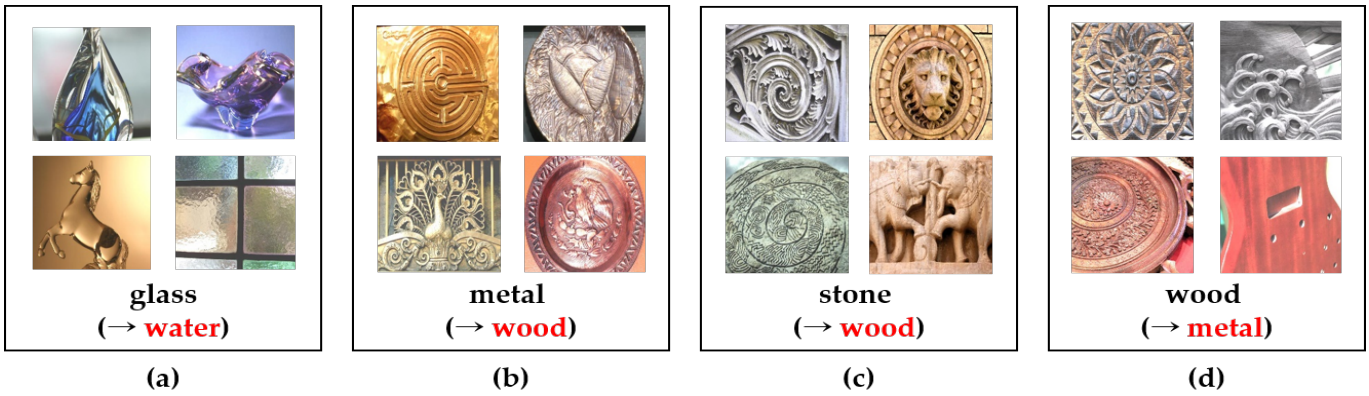

Figure 8. Examples of images in FMD for a class selected according to the result of the confusion matrix. (**a**–**d**) mean class glass, metal, stone, and wood examples, respectively. The black letter below the image indicates actual class, and the red letter indicates predicted class by the proposed model.

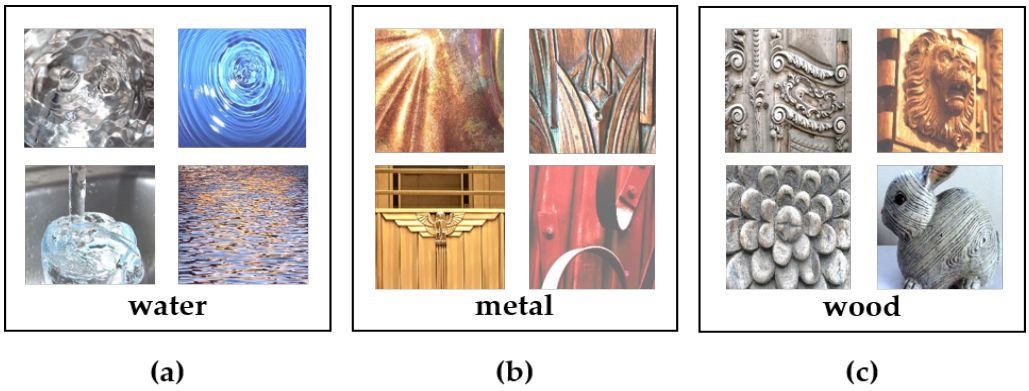

Figure 9. Examples of class images in FMD. (**a**–**c**) are class water, metal, and wood examples, respectively.

4.5. Ablation Study

We conducted an ablation study to verify the effectiveness of FRM. We compared the performance of DenseNet161, which was used as the backbone, and the model presented in this study to confirm its effectiveness. Table 2 lists the experimental results for the texture datasets. Block 14 is shown in Figure 6a. Its structural characteristic is the form that FRM combines with each DenseBlock. It has all the feature information from low to high level. Comparing the accuracy results to the backbone, Block 14 showed +3.96%, 3.09%, and +11.08% accuracy for DTD, FMD, and KTH-TIPS2b, respectively. The results proved that the combination of the FRM with the backbone model is effective for classification performance. FRM consists of a combination of local parts using convolutions of different sizes and global parts using average values, which can help the model determine the properties of the unique patterns of images in the dataset. That is, it was confirmed that FRM is useful in improving performance for datasets showing regularity and irregularity characteristics. In particular, it showed that it is effective at finding characteristics in images for KTH-TIP2b that have regularity.

Table 2. Comparison of the results of the backbone and proposed model on datasets.

Method	DTD	FMD	KTH-TIP2b
DenseNet161 (Backbone)	70.57	79.67	81.89
Block 14 (Ours)	74.53	82.76	92.97

The backbone has a structure that minimizes the information loss as the model deepens. However, this was limited to classifying the detailed patterns of texture images and improving their performance. Therefore, the experimental results of structures that combine FRM with the backbone, such as Block 14, showed improved performance. Moreover, it showed that the FRM assisted in identifying and classifying image patterns of DTD, FMD, and KTH-TIP2b in terms of effectiveness.

5. Extended Study

In this section, we introduce additional studies to identify improved modules suitable for models that combine various features from low to high levels. The model structure shown in Figure 6 was maintained, and an FRM with some changes in the internal structure was used. We evaluated whether the changed FRM affected suitability and performance when combined with the model structures shown in Figure 6.

5.1. Overview of Modified Module

Figure 10 shows the two modified versions of the proposed module. The conditions for using filters of different sizes were the same as those used for the proposed modules, as shown in Figure 3. However, there are structural differences. As shown in Figure 10a, it is a structure in which some conv 1×1 is removed from the proposed FRM in Figure 3, inspired by the inception module structure with the basic structure presented in Figure 4a. Figure 10b shows a method of using the output value directly instead of a structure in which the output value from DenseBlock is divided into local and global values separately. Moreover, this method did not use a max-pooling layer.

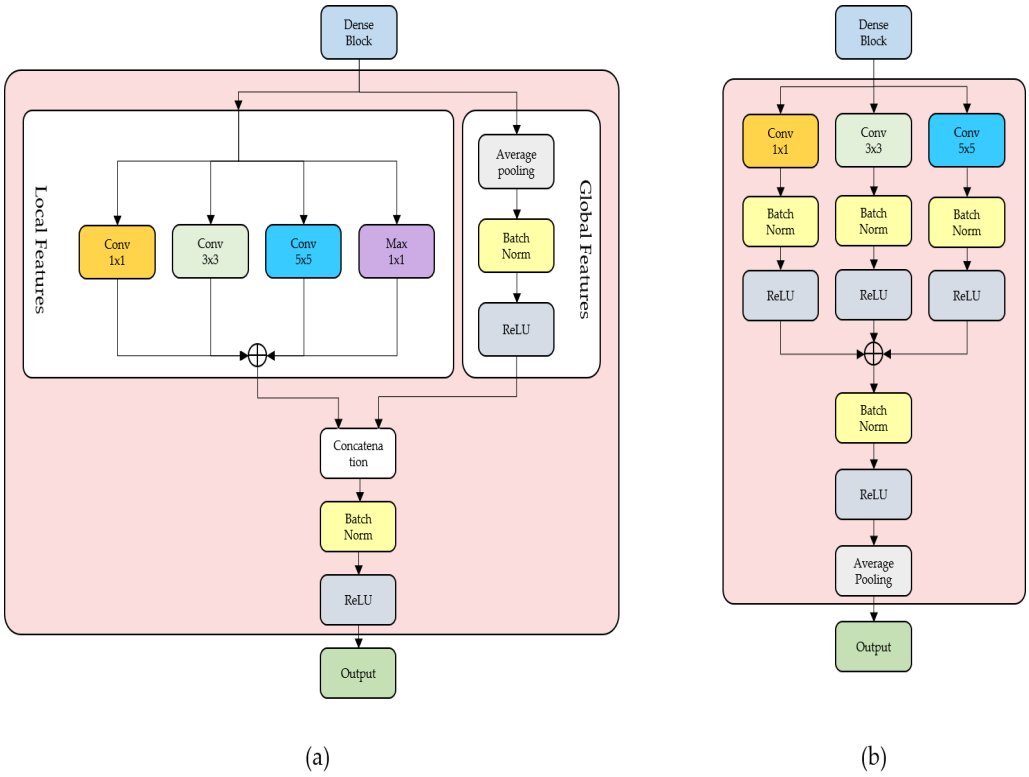

Figure 10. (**a**,**b**) show the structure of the changed module.

5.2. Experiment Result

We compared the performance of the FRM with that of the changed modules, as shown in Figure 10. Table 3 lists the results of the three methods for the texture datasets. Method 1 denotes the case in which the proposed FRM is used, as shown in Figure 3. Methods 2 and 3 refer to the combination of the modified modules proposed in Figure 10a and Figure 10b,

respectively. The blocks mentioned in each method refer to the structural forms shown in Figure 6. Method 1 was demonstrated to be effective using the proposed method with the FRM, as described in Section 4.3. Method 2 exhibited high performance when using high-level features across the entire dataset. Method 3 performed well when several levels were used in DTD and FMD, whereas it performed better only at high levels in KTH-TIPS2b. A comparison of the performance results for all methods is shown in Figure 11. Overall, the results for the DTD showed the highest performance of 75.50% when Block 24 in Method 3 was used. In the FMD results, Block 14 in Method 3 was the highest at 83.79%. Block 14 in Method 1 for KTH-TIP2b exhibited a performance of 92.97%. In addition, improvements of 0.85% and 0.57% were observed for Block 4 in Methods 2 and 3, respectively.

Table 3. Experimental results on texture datasets. Bold denotes best accuracy within each module method from Blocks 14 to 4. Method 1 refers to models that combine the proposed modules. Methods 2 and 3 refer to models using the modules mentioned in Figure 10a and 10b, respectively.

Method		Backbone	DTD	FMD	KTH-TIP2b
Method 1	Block 14	DenseNet161	**74.53**	**82.76**	**92.97**
	Block 24	DenseNet161	74.52	82.07	91.55
	Block 34	DenseNet161	74.35	81.72	91.83
	Block 4	DenseNet161	74.35	81.72	91.76
Method 2	Block 14	DenseNet161	74.83	82.76	91.19
	Block 24	DenseNet161	74.47	82.07	91.48
	Block 34	DenseNet161	**74.89**	82.41	91.76
	Block 4	DenseNet161	74.71	**83.45**	**92.12**
Method 3	Block 14	DenseNet161	75.26	**83.79**	91.62
	Block 24	DenseNet161	**75.50**	81.03	91.97
	Block 34	DenseNet161	75.14	81.72	92.26
	Block 4	DenseNet161	74.77	82.41	**92.40**

In this section, the experiments demonstrated that the modified modules performed better than the proposed modules on the datasets except for KTH-TIP2b. Furthermore, based on these experimental results, the performance improvement factors of the changed module are as follows. The feature of DenseBlock is used immediately without using the conv 1×1 filter used for dimension reduction in the proposed module. And the features are used for convolutional filters of various sizes. This is a common structural feature of Figure 10a,b. In addition, there is also a simplified structural form of the module shown in Figure 10b. Thus, in the future, we should develop modules based on these performance enhancement factors and conduct research to find other enhancement factors. Moreover, further studies should be conducted to overcome the performance limitations of FRM.

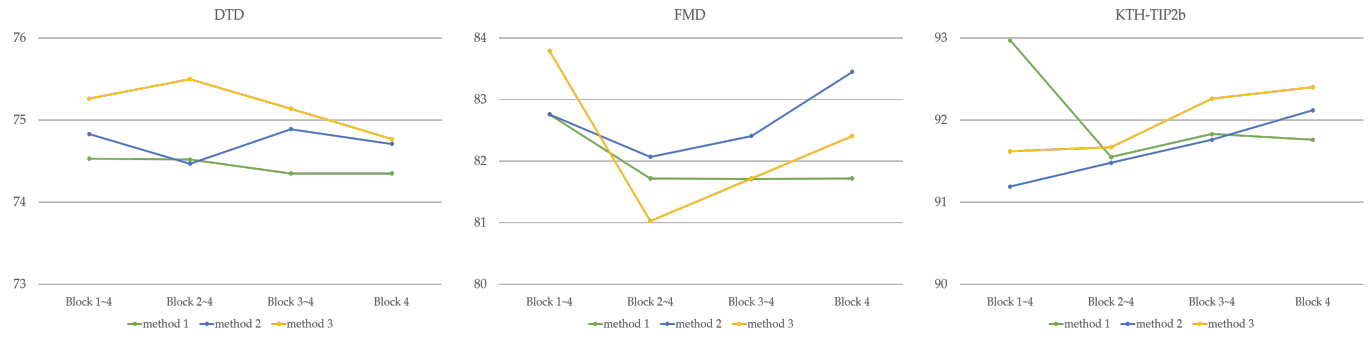

Figure 11. The graph of evaluation results for each dataset.

6. Conclusions

In contrast to object-based datasets, texture datasets consist of images that are closely or partially cropped around the unique pattern of a particular material. To analyze and evaluate datasets with these features, we proposed a model that comprehensively used low-to-high-level features to distinguish subtle pattern differences in each class. In addition, an FRM was combined with DenseBlock in the model to identify and maintain significant features from each DenseBlock. To demonstrate the effectiveness of the proposed model, an evaluation was conducted using several structural combinations. The experimental results showed that a structure that aggregated multiple features could perform effectively. However, a comparative experiment between the proposed FRM and modified module structure in an extended study demonstrated structural limitations in the FRM performance. The results of this study highlight the necessity for further study to identify improved modules suitable for the proposed structures in the future. Moreover, to increase the utilization of the proposed model in the study, experimental studies on datasets with various characteristics other than texture datasets should be conducted. Further research is also needed on whether the proposed model is suitable for segmentation and object detection.

Author Contributions: Methodology, S.-H.P. and S.-Y.A.; investigation, S.-H.P., S.-Y.A. and S.-W.L.; software, S.-H.P. and S.-Y.A.; writing, S.-H.P. and S.-Y.A.; review, S.-W.L.; supervision, S.-W.L.; and funding acquisition S.-W.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Gachon University research fund of 2020(202008450003) and the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) (No. RS-2023-00250978).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets used in this study are available at the following links: https://www.robots.ox.ac.uk/~vgg/data/dtd/index.html (accessed on 8 September 2023), https://people.csail.mit.edu/lavanya/fmd.html (accessed on 8 September 2023), https://www.csc.kth.se/cvap/databases/kth-tips/credits.html (accessed on 8 September 2023).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liu, L.; Chen, J.; Fieguth, P.; Zhao, G.; Chellappa, R.; Pietikäinen, M. From BoW to CNN: Two decades of texture representation for texture classification. *Int. J. Comput. Vis.* **2019**, *127*, 74–109. [CrossRef]
2. Liu, L.; Fieguth, P.; Guo, Y.; Wang, X.; Pietikäinen, M. Local binary features for texture classification: Taxonomy and experimental study. *Pattern Recognit.* **2017**, *62*, 135–160. [CrossRef]
3. Fradi, H.; Fradi, A.; Dugelay, J.L. Multi-layer Feature Fusion and Selection from Convolutional Neural Networks for Texture Classification. In Proceedings of the VISIGRAPP (4: VISAPP), Vienna, Austria, 8–10 February 2021 ; pp. 574–581.
4. Chen, Z.; Li, F.; Quan, Y.; Xu, Y.; Ji, H. Deep texture recognition via exploiting cross-layer statistical self-similarity. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 5231–5240.
5. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
6. Krizhevsky, A.; Hinton, G. *Learning Multiple Layers of Features from Tiny Images*; Technical Report; University of Toronto: Toronto, ON, Canada, 2009.
7. Wah, C.; Branson, S.; Welinder, P.; Perona, P.; Belongie, S. *The Caltech-Ucsd Birds-200-2011 Dataset*; Technical Report; California Institute of Technology: Pasadena, CA, USA, 2011.
8. Cimpoi, M.; Maji, S.; Kokkinos, I.; Mohamed, S.; Vedaldi, A. Describing textures in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 3606–3613.
9. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
10. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
11. 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, NV, USA, 27–30 June 2016; pp. 770–778.

12. 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, HI, USA, 21–26 July 2017; pp. 4700–4708.
13. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
14. Chai, J.; Zeng, H.; Li, A.; Ngai, E.W. Deep learning in computer vision: A critical review of emerging techniques and application scenarios. *Mach. Learn. Appl.* **2021**, *6*, 100134. [CrossRef]
15. Guo, Y.; Liu, Y.; Oerlemans, A.; Lao, S.; Wu, S.; Lew, M.S. Deep learning for visual understanding: A review. *Neurocomputing* **2016**, *187*, 27–48. [CrossRef]
16. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A comprehensive survey on transfer learning. *Proc. IEEE* **2020**, *109*, 43–76. [CrossRef]
17. Dai, X.; Yue-Hei Ng, J.; Davis, L.S. Fason: First and second order information fusion network for texture recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7352–7360.
18. Hu, Y.; Long, Z.; AlRegib, G. Multi-Level Texture Encoding and Representation (Multer) Based on Deep Neural Networks. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 4410–4414. [CrossRef]
19. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
20. Csurka, G.; Dance, C.; Fan, L.; Willamowski, J.; Bray, C. Visual categorization with bags of keypoints. In Proceedings of the Workshop on Statistical Learning in Computer Vision, ECCV, Prague, Czech Republic, 11–14 May 2004; Volume 1, pp. 1–2.
21. Sivic.; Zisserman. Video Google: A text retrieval approach to object matching in videos. In Proceedings of the Ninth IEEE International Conference on Computer Vision, Nice, France, 13–16 October 2003; pp. 1470–1477.
22. Vasconcelos, N.; Lippman, A. A probabilistic architecture for content-based image retrieval. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662), Head Island, SC, USA, 13–15 June 2000; Volume 1, pp. 216–221.
23. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
24. Bay, H.; Tuytelaars, T.; Van Gool, L. Surf: Speeded up robust features. In Proceedings of the Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, 7–13 May 2006; Proceedings, Part I 9; Springer: Berlin/Heidelberg, Germany, 2006; pp. 404–417.
25. Bruna, J.; Mallat, S. Invariant scattering convolution networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1872–1886. [CrossRef] [PubMed]
26. Chan, T.H.; Jia, K.; Gao, S.; Lu, J.; Zeng, Z.; Ma, Y. PCANet: A simple deep learning baseline for image classification? *IEEE Trans. Image Process.* **2015**, *24*, 5017–5032. [CrossRef] [PubMed]
27. Jolliffe, I. *Principal Component Analysis*; Springer: New York, NY, USA, 2002.
28. Sharan, L.; Rosenholtz, R.; Adelson, E. Material perception: What can you see in a brief glance? *J. Vis.* **2009**, *9*, 784–784. [CrossRef]
29. Caputo, B.; Hayman, E.; Mallikarjuna, P. Class-specific material categorisation. In Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, Beijing, China, 17–21 October 2005; Volume 2, pp. 1597–1604.
30. Lin, T.Y.; Maji, S. Visualizing and understanding deep texture representations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2791–2799.
31. Cimpoi, M.; Maji, S.; Vedaldi, A. Deep filter banks for texture recognition and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3828–3836.
32. Song, Y.; Zhang, F.; Li, Q.; Huang, H.; O'Donnell, L.J.; Cai, W. Locally-transferred fisher vectors for texture classification. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 4912–4920.
33. Zhang, H.; Xue, J.; Dana, K. Deep ten: Texture encoding network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 708–717.
34. Bu, X.; Wu, Y.; Gao, Z.; Jia, Y. Deep convolutional network with locality and sparsity constraints for texture classification. *Pattern Recognit.* **2019**, *91*, 34–46. [CrossRef]