A ROBUST-TEXTURE CONVOLUTIONAL NEURAL NETWORK

***Walairach Nunsong*[1], *Kuntpong Woraratpanya*[2]**

[1,2]Faculty of Information Technology, King Mongkut's Institute of Technology Ladkrabang, Chalongkrung Rd. Ladkrabang, Bangkok, Thailand 10520

E-mail: o pu@hotmail.com[1], kuntpong@it.kmitl.ac.th[2]

DOI: https://doi.org/10.22452/mjcs.sp2019no2.9

ABSTRACT

AlexNet was a breakthrough for the convolutional neural network (CNN) and showed the greatest successful modified CNN that works well with large-scale images. However, it was unsuccessful in texture classification tasks. To extend CNN's capability, this paper proposes a modified CNN architecture called a robust-texture convolutional neural network (RT-CNN) to serve both complex shape and texture classification tasks, especially in the following challenges: (i) the same class of images naturally contains various viewpoints, scales, uneven illuminations, etc.; (ii) similarly shaped objects with different textures of images are often assigned into different classes; and (iii) different shaped objects with similar textures of images are often assigned into the same class. The proposed scheme embeded a texture-embedded supplementary method, composed of texture compensation and supplement, into the CNN architecture. The texture compensation is constructed from texture subbands decomposed by 2D Littlewood-Paley empirical wavelet transform (2D Littlewood-Paley EWT). Then the texture supplement is constructed from texture subbands by using Gabor wavelet to extract multi-scale and multi-orientation texture features. Based on two challenging datasets, the experimental results show that RT-CNN outperforms all test baseline methods: AlexNet, T-CNN, and wavelet-CNN, in terms of recognition accuracy rate. On a typical dataset, the recognition accuracy rate of the proposed method is still better than those of T-CNN and wavelet-CNN, and is comparable to that of AlexNet.

Keywords*: *RT-CNN, convolutional neural network, robust texture, texture subband, texture classification

1.0 INTRODUCTION

A convolutional neural network (CNN) is a feed-forward artificial neural network. The classical CNN is LeNet-5 proposed by LeCun et al. in 1998 [1]. It was successfully applied to handwritten digit recognition. The prominent architecture of CNN is that feature extraction and classification are cascaded in a single pipeline. Inside feature extraction and classification are convolutional layers and fully connected layers, respectively. This architecture makes it robust to translation, scaling, rotation, and noise [2]. This CNN architecture was very attractive to re- searchers at that time. Since then, many modified CNNs have been proposed for a variety of applications, such as face detection and recognition [3], [4], Chinese license plate recognition [5], and micro nucleus in human lympho- cyte image detection [6]. However, these modified architectures worked well with small-scale images. In 2012, it was a breakthrough for CNN when AlexNet, introduced by Krizhevsky et al. [7], showed the greatest successful modified CNN that worked well with large-scale images. AlexNet is a deep convolutional neural network archi- tecture that shows its ability in classifying the 1.2 million high-resolution images in the ImageNet dataset. It can also alleviate the overfitting problem, which always occurs in large-scale images, by means of data augmentation and dropout. For these reasons, AlexNet becomes the state of the art for various machine vision tasks [8], [9], [10]. Although the conventional CNN architecture achieves overall performance in object/shape analysis as reported in [11], [12], [13], [14], its performance in texture analysis is still challenging due to the following cases: (i) the same class of images naturally contains various viewpoints, scales, uneven illuminations, etc. [15]; (ii) similarly shaped objects with different textures of images are often assigned into different classes [16]; and (iii) different shaped objects with similar textures of images are often assigned into the same class [16]. These characteristics of texture images are different from those of object images. In addition, the conventional CNN extracts features based on a deep convolutional architecture to represent a shape of an object. However, the texture loss is occurred when the deeper convolution is used [17]. Responding to these challenges, modified CNNs for texture classification have

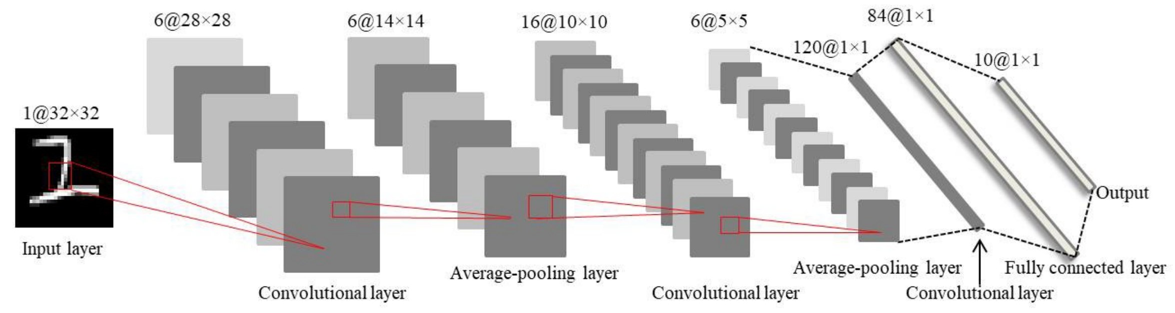

Fig. 1: Architecture of LeNet-5

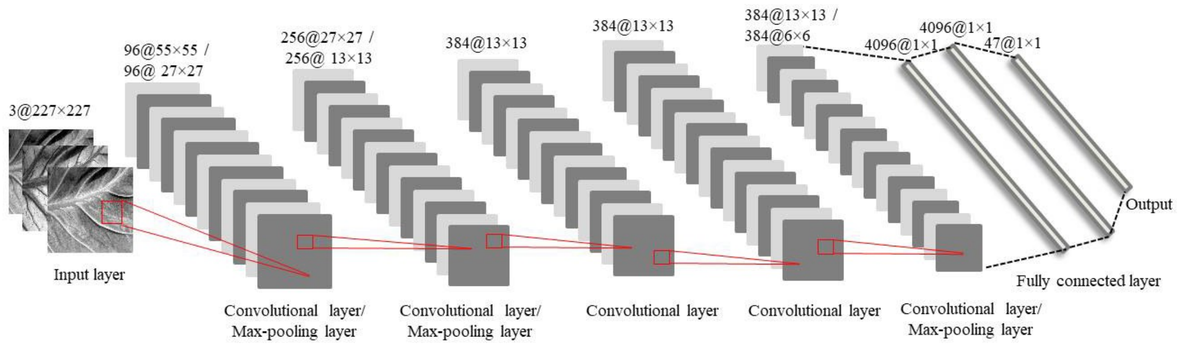

Fig. 2: Architecture of AlexNet

been proposed to improve their performance. To the best of our knowledge, CNNs for solving texture analysis can roughly be categorized into three solutions: (i) data augmentation, (ii) modified CNN architecture, and (iii) a combination of data augmentation and modified CNN architecture. For the first solution, the performance of CNN can be improved by using a data augmentation technique. Data augmentation is a process that splits a large image into a set of small images called patches. This process increases the number of input images for training the CNN model based on a variety of input patterns. It can reduce the overfitting problem [7]. For example, Luiz G. et al. [18] presented a CNN model based on texture analysis for high resolution images in a forest species dataset. Luiz's method applied a data augmentation technique to break down a high resolution image into a set of patches. The data augmentation was applied to both training and testing processes. With the augmentation technique, the diversity of patterns is increased for CNN training; thus, the recognition accuracy rate is increased in CNN testing. Eunsoo P. et al. [19] proposed a CNN model based on fingerprint patch extraction. Eunsoo's data augmentation generated patches from fingerprint image by using rotation and random cropping. This technique can reduce the effect of the overfitting problem. In addition, the model trained on patches from Eunsoo's data augmentation was robust to image rotation. Then all fingerprint patches were applied to training and testing processes of CNN. Wang Q. et al. [20] developed a CNN model to recognize high resolution computed tomography (CT) images in a lung dataset. They decomposed a high resolution image into a set of patches at different scales and orientations by means of Gabor and local binary pattern (LBP). This makes the CNN model robust to scale, orientation, and rotation of images.

Although data augmentation can improve the performance of CNN in terms of recognition accuracy rate, its implementation scheme cannot be done within a single model. In order to achieve a simple architecture under the single model, modification of CNN architecture for texture analysis is proposed as the second solution for solving texture analysis based on CNN. Texture CNN (T-CNN) proposed by Andrearczyk et al.[17] was based on the use of filter banks in CNN. It was developed under the concept of preserving texture. In this concept, an energy feature vector is generated by averaging each feature map of the last convolutional layer. This energy feature vector is called an energy layer. The energy layer is inserted between the last convolutional layer and the first fully connected layer. This leads to a reduction of parameters, memory requirement, and computation time. However, the energy layer always contains only the low-frequency features, but not high-frequency features [21]. In other words, the high-frequency features are lost during the training process. In order to solve the texture loss and overfitting problems, a combination of data augmentation and modified CNN architecture is introduced to solve the texture analysis as the third solution. S. Fujieda et al. [21] developed a wavelet CNN to solve both problems by using data

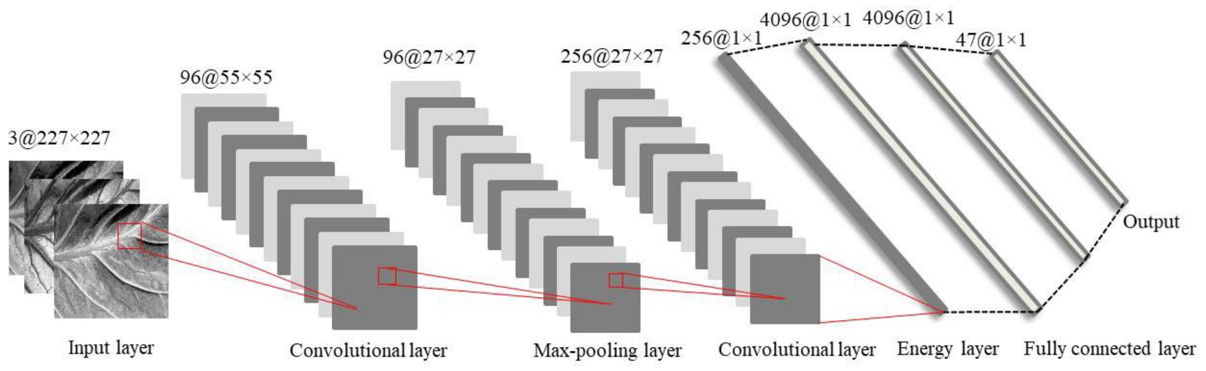

Fig. 3: Architecture of T-CNN with two convolutional layers (T-CNN-2)

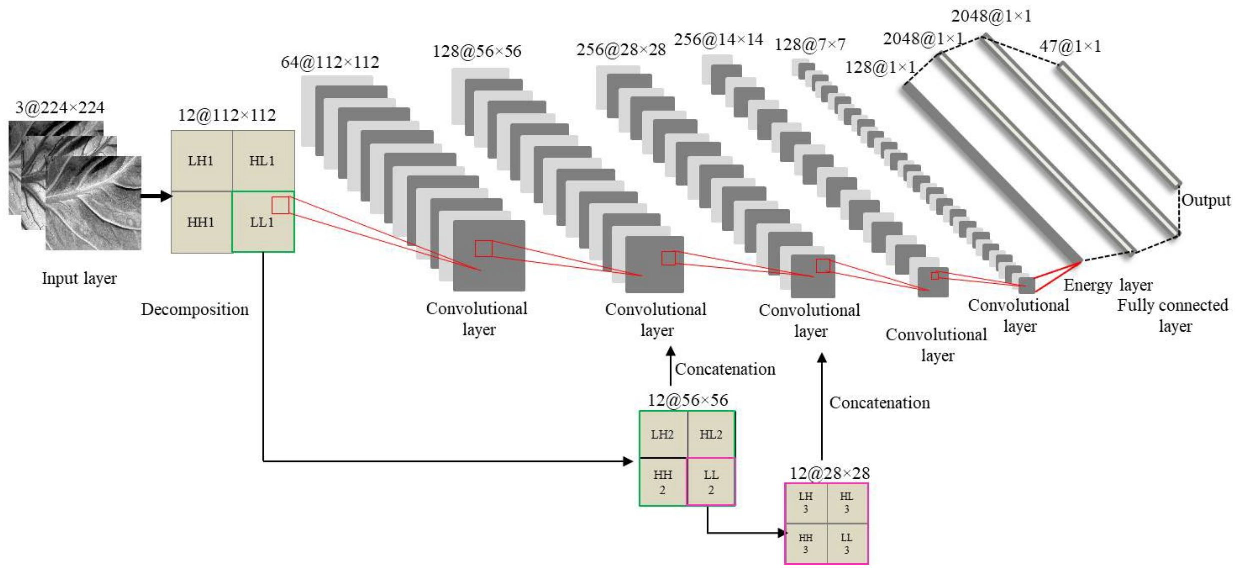

Fig. 4: Architecture of wavelet CNN with 3-level decomposition

augmentation and wavelet transform. Ordinarily, the conventional CNN can capture features well based on spatial domain, but cannot capture features based on spectral domain. Nevertheless, scale invariant features commonly provide in spectral domain. Therefore, the concept of the wavelet CNN is a combination of the features from both spatial and spectral domains under a single model. The wavelet CNN not only reduces the overfitting problem, but also reduces the information loss. However, wavelet decomposition can extract only features from four subbands containing one low and three high frequencies. In fact, texture images not only consist of low and high frequencies, but also include multiple scales with uncertain directions of texture. This means that the wavelet decomposition cannot efficiently preserve significant texture information. Certainly, the performance improvement of CNN is still challenging in texture analysis. When we look inside each convolutional layer, all filters are optimized by a backpropagation algorithm, leading to optimized filter construction from real data. However, for compensation of the lost texture, adaptive texture filters are essential. For this reason, this paper proposes a modified CNN architecture called a robust-texture convolutional neural network (RT-CNN). The remainder of this paper is organized as follows. Architectures of the conventional CNN and of modified CNNs are reviewed and research problems are formulated in Section 2.0. Section 3.0 introduces our proposed RT-CNN method. The experimental results are reported and discussed in Section 4.0, and finally, the conclusion is presented in Section 5.0.

2.0 PROBLEM FORMULATION

In this section, we focus on a modified CNN architecture solution in order to keep the concept of a single pipeline. Architectures of the conventional CNN and of modified CNNs are reviewed and their limitations are pointed out below.

2.1 Convolutional Neural Network Architecture and Its Extensions

A conventional CNN architecture originates from the feature extraction cascaded with classification. As schematically shown in Fig. 1, the feature extraction consists of multiple pairs of layers, where a pair of layers contains a convolutional layer and an average-pooling layer. The conventional CNN architecture consists of multiple pairs of layers for feature extraction, except that the last convolutional layer is single. The convolutional layer extracts different features from receptive fields, which are input images. The outputs of the convolutional layer are feature maps. The average-pooling layer cascaded from the convolutional layer reduces the size of feature maps by applying an average filter of size 2×2 with stride of 2 to sub-regions of the feature map. This operation is repeated from one layer to the next layer over and over. In feature extraction, simple features, such as straight edges, colors, and curves, are extracted by the first convolutional layer, whereas complex features, such as the shape of an object, are extracted by the last convolutional layer. After that, all extracted features are classified by a fully connected layer which is also known as a classifier. The conventional CNN uses two fully connected layers for activating complex features to obtain the overall shape of the image. The outputs of the last fully connected layer are probabilities for each class. The maximum probability is always assigned to the predictive class. AlexNet [7] is a deep convolutional neural network that, in practice, provides excellent results in object analysis. Its architecture contains five convolutional layers. The early convolutional layers are usually followed by max-pooling layers. A max filter of size 2×2 with stride of 2 is applied to sub-regions of the feature map and then the maximum value of each sub-region is selected. The three fully connected layers are used for classification as schematically shown in Fig. 2. A softmax function is applied for calculating probabilities of the last fully connected layer. AlexNet's architecture can also prevent an overfitting problem by using a dropout technique [22] in the fully connected layers. However, AlexNet does not work well in texture analysis, since the early convolutional layer plays a major role in low-pass filters, thus most textures are filtered out from this operation as illustrated in Subsection 2.2. A texture convolutional neural network (T-CNN) [17], extended from AlexNet, can work well in texture analysis. It is mainly designed for reducing the complexity in terms of the number of parameters. As a result, less memory and computing time are consumed. As schematically shown in Fig. 3, T-CNN architecture is modified in a gap of feature extraction and classification such that an energy layer is inserted between the last convolutional layer and the first fully connected layer. The energy layer pools energy of the feature maps of the last convolutional layer. Each feature map is calculated by averaging its activated outputs. The output vector of the energy layer is forwarded to the first fully connected layer. The output probabilities are calculated using the softmax function. T-CNN usually provides good performance when three convolutional layers are used. However, most textures are lost in the output vector of the energy layer extracted from the feature maps. This is a reason why T-CNN cannot achieve a great performance improvement. A wavelet CNN [21] is a modified architecture constructed from a combination of a conventional CNN and a wavelet spectral analysis. As schematically depicted in Fig. 4, the wavelet spectral analysis is used to decompose an image into low-frequency and high-frequency subbands. The outcomes of this stage are disseminated to the regular convolutional layer path and the compensation path; that is, all decomposed subbands are forwarded to the regular convolutional layers while the decomposed low-frequency subband is concatenated to feature maps before forwarding to the next convolutional layer. The modified architecture uses convolutional filters with stride of 2 and 1×1 padding. This can reduce the size of feature maps and remove the max-pooling layers without diminishing the recognition accuracy rate. An energy layer [17] is added to improve the performance on the small number of parameters. This approach requires data augmentation for generating training sets. In this way, an input image is randomly cropped and flipped to generate a variety of training images. The wavelet CNN provides the best result when four convolutional layers with four level decompositions are used. However, data augmentation is still necessary.

2.2 Analysis of CNN Architecture for Texture Classification

This subsection aims to analyze the problems of modified CNN architectures for texture classification and to point out the main causes of lower performance.

AlexNet is a well-designed architecture for object analysis. It can efficiently extract a shape or an external boundary of an object. This success comes from of layer to layer convolution operation; that is, the last convolutional layer can extract complex features from simple features of the early convolutional layer. The complex features, which always represent overall shapes of images, are forwarded to a fully connected layer portion. The classification results in the last fully connected layer in the form of probabilities of predictive classes [17]. Although AlexNet provides excellent results in object detection and recognition tasks, it does not achieve a good performance in texture analysis. The main cause is texture loss. The texture is easily lost due to a deep convolutional architecture

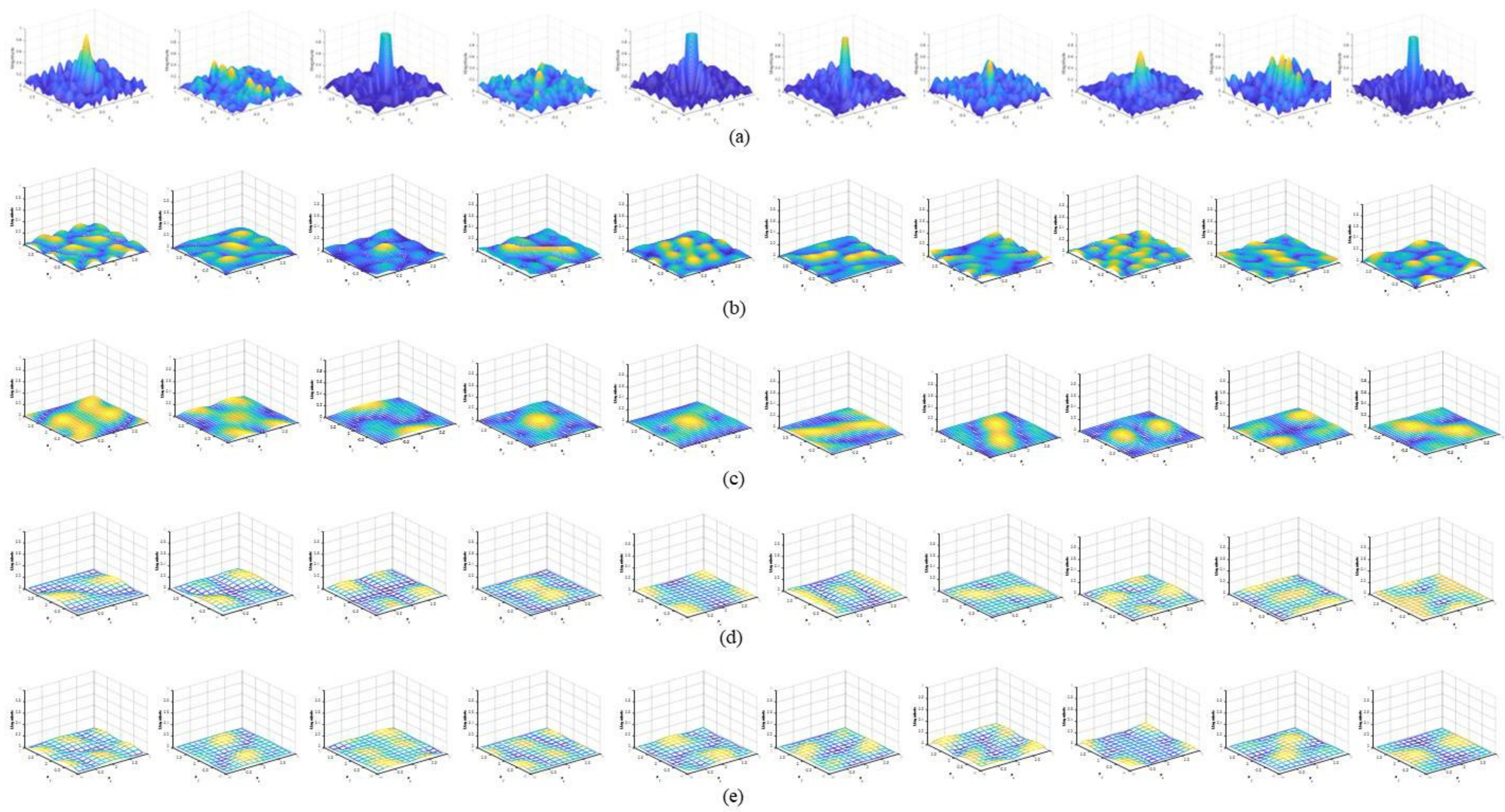

Fig. 5: Examples of filters determined by AlexNet architecture. Figs. (a), (b), (c), (d), and (e) are a set of filters of the first, second, third, forth, and fifth convolutional layers, respectively.

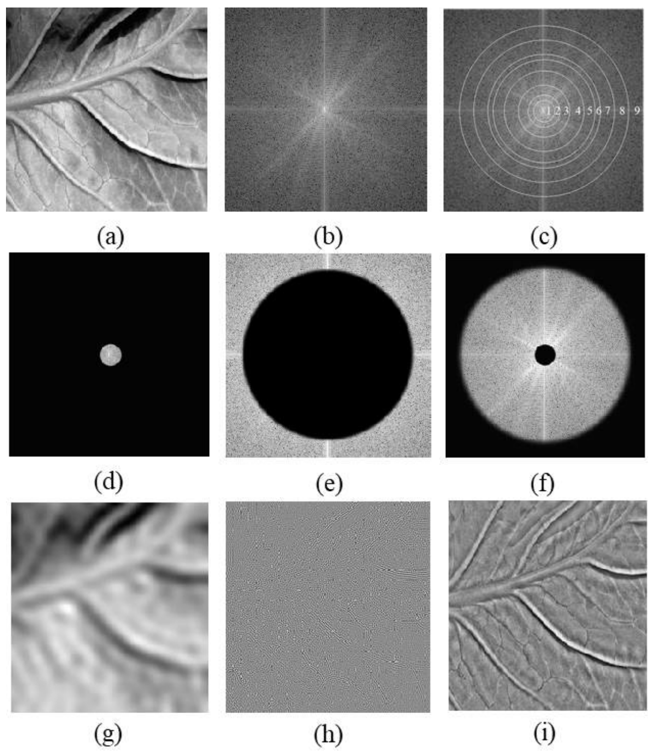

Fig. 6. Texture component decomposition. (a) An original image, (b) a Fourier spectrum of Fig. 6(a), (c) a segmented Fourier spectrum, (d) a shape spectrum, (e) a noisy spectrum, (f) a texture spectrum, (g) a shape component, (h) a noisy component, and (i) a texture component.

from layer to layer of the conventional CNN. As shown in Fig. 2, when we look inside a feature extraction portion, all feature maps are extracted by filters of convolutional layers, where these filters are optimized from real data. This results in well featured maps. In another point of view, the deep convolutional architecture commonly makes the texture lost, since CNN extracts features in the form of multiresolution analysis as reported in [21]; that is, the deeper convolutional architecture, the higher texture loss. This phenomenon can be seen in Fig. 5 such that all optimized filters are in Fourier domain. A sample of the first convolutional layer as shown in Fig. 5(a) looks similar to a low-pass filter. This means that this kind of filter allows the low frequency passing through the next layer, but it blocks most median and high frequencies. In this case, the most textures are lost at this layer. When we look into deeper layers, the second, third, forth, and fifth convolutional layers are similar to high band pass filters as shown in Figs. 5(b), 5(c), 5(d), and 5(e), respectively. These filters allow most median and high frequencies passing through the feature map of the first convolutional layer. However, the most median and high frequencies in feature map are already lost. Therefore, in AlexNet's architecture containing five convolutional layers, the last layer always represents the low-frequency feature. Indeed, the texture possibly consists of both median and high frequencies. This results in the low performance for texture classification.

A texture CNN (T-CNN) was proposed to alleviate the texture loss problem. The T-CNN used a shallow convolutional architecture. It uses only three convolutional layers, but adds an energy layer. The role of the energy layer is to extract feature maps from the last convolutional layer and to present the result in one value for one feature map. This shallow convolutional architecture with an energy layer added not only preserves image texture but also reduces the overfitting problem which occurs from a large number of parameters. Indeed, the achievement of T-CNN is parameter reduction. However, it is not clear that the T-CNN can improve the performance by texture preservation. When we look inside the complex feature, it still uses the low-frequency feature that can preserve some textures for classification. To solve the texture loss problem, a wavelet CNN was proposed to overcome this problem. This architecture can preserves both low and high frequencies. In other words, it is a deep convolutional architecture that can preserve texture. This approach applies wavelet transform for decomposing low and high frequencies of images. These decomposed images instead of the whole image (undecomposed image) are used for training the model. This can reduce median and high frequency loss when the input image is extracted feature map by the first convolutional layer. In addition, each level of decomposed images is concatenated with feature maps of CNN before forwarding to the next convolutional layer except the first convolutional layer. This architecture can alleviate the texture loss problem, when the feature extraction portion is deepened. However, the wavelet CNN can preserve only some textures based on the wavelet transform that can decompose an image into four subbands: low-low, low-high, high-low, and high-high subbands [23], it cannot preserve other details such as scales and directions of the textures. As mentioned, the problem of conventional and modified CNN architectures for texture analysis is texture loss. The main causes can be summarized as: (i) the deeper convolutional architecture and (ii) the use of undecomposed images for training.

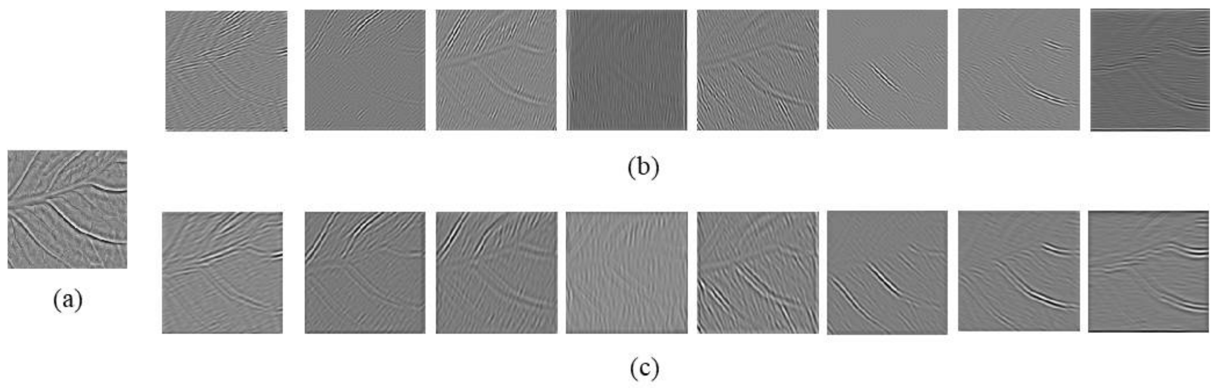

Fig. 7: Robust texture descriptors. (a) a texture component, (b) a set of the first scale with eight orientations of Fig. 7(a), and (c) a set of the second scale with eight orientations of Fig. 7(a).

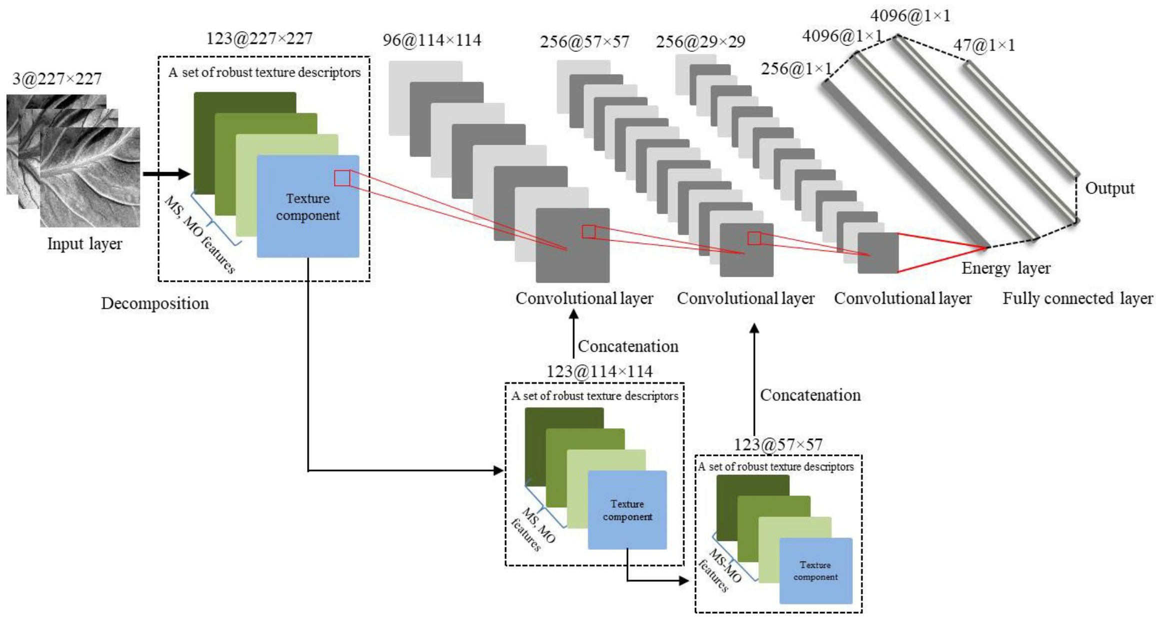

Fig. 8. Architecture of robust-texture convolutional neural network with three convolutional layers.

3.0 PROPOSED METHOD

This paper aims to propose a robust-texture convolutional neural network for texture classification. Concepts and algorithms of the proposed method are covered in the following topics: definitions of texture component and a set of robust-texture descriptors, architecture of a robust-texture convolutional neuron network, and a robust texture-embedded supplementary method.

3.1 Definitions of Texture Component and a Set of Robust-Texture Descriptors

A texture component and robust-texture descriptors of an image can be defined as below.

Definition 1: Let F be a Fourier spectrum of an image f. If F is adaptively segmented into N non-overlapping ring-shape subbands and then the first and last subbands are removed, the remaining subbands are texture subbands.

Definition 2: Let B and W be the texture component from *Definition 1* and the texture filters defined by v scales and u orientations, respectively, in spatial domain. The convolution of B and W generates multi-scale and multi-orientation features, G. The concatenation of B and G yields a set of robust-texture descriptors.

From *Definition 1*, a visual definition can be expressed by using Veined image in DTD dataset as shown in Fig. 6(a). Fourier spectrum F of Fig. 6(a) is shown in Fig. 6(b). In order to decompose a texture component from an image, f, F is segmented into N non-overlapping ring-shape subbands by using a 2D Littlewood-Paley empirical wavelet transform (2D Littlewood-Paley EWT) based on local minima detection with scale-space histogram segmentation [24], [25]. This method not only adaptively segments spectrum but also efficiently decomposes meaningful components. Fig. 6(c) shows that Fourier spectrum F of Fig. 6(b) is segmented by using the 2D Littlewood-Paley EWT. It is decomposed into 9 subbands labeled with numbers 1 to 9. A characteristic of segmentation in Fig. 6(c) is ring-shape. According to *Definition 1*, the first and last subbands of Fig. 6(c) contain shape and noisy spectra as depicted in Figs. 6(d) and 6(e), respectively. F without the first and last subbands as shown in Fig. 6(f) is a texture spectrum corresponding to *Definition 1*. Figs. 6(g), 6(h), and 6(i) are inverse Fourier transforms of Figs. 6(d), 6(e), and 6(f), respectively. Fig. 6(g) shows the blurred shape image corresponding to the main object in the original image. Grainy image, i.e., a noise component is depicted in Fig. 6(h) which greatly differs from the original image. Fig. 6(i) shows texture and sharp edge—a texture component of the original image. Moreover, the structure of Fig. 6(i) is the similar to that of Fig. 6(a). This strongly supports that the texture component obtained from local minima spectrum segmentation by using scale-space histogram satisfies our *Definition 1*. From *Definition 2*, a visual definition can be expressed by using a texture component image derived from *Definition 1* as shown in Fig. 7(a). When the texture component B is filtered by Gabor wavelet with v scales and u orientations, the multi-scale and multi-orientation features are generated as illustrated in Figs. 7(b) and 7(c), respectively. A set of robust-texture descriptors are formed from a concatenation of Figs 7(a), 7(b), and 7(c). This method not only makes use of the multi-scales and multi-orientations to decompose the significant textures but also helps remove noise. This strongly proves that a set of robust-texture descriptors obtained from extracting multi-scale and multi-orientation features by using Gabor wavelet and then concatenated with the texture component satisfies our *Definition 2*. According to *Definitions 1* and *2*, a set of robust-texture descriptors are important for representing complete image contents. Moreover, these texture descriptors make the model robust to various scales and orientations of texture in images. Therefore, the proposed method uses a robust texture-embedded supplementary method to construct the robust-texture descriptors for supporting texture preservation. The robust texture-embedded supplementary section is embedded into CNN under a single pipeline. The embedded architecture is called a robust-texture convolutional neural network (RT-CNN). More details of the proposed architecture are given below.

3.2 Architecture of Robust-Texture Convolutional Neural Network

From distinctive point of CNN, all filters of convolutional layers are automatically optimized by training process. This leads to create optimized filters for feature extraction of CNN. Typically, filters of the early convolutional layer are trained from input image contents such as edge, shape, and texture, and then are forwarded to the next convolutional layer. In other words, the filters of the next convolutional layer are trained from the feature map of its previous convolutional layer. As mentioned in Section 2.0, the image contents in the feature map of the first convolutional layer are lost. Therefore, compensation of the lost texture is essential. Responding to this, the robust-texture convolutional neural network is proposed. The robust texture-embedded supplementary section is added to the CNN model for providing a set of robust-texture descriptors. This supplementary section composed of a set of robust-texture descriptors is embedded in front of the first convolutional layer and also concatenated with the feature map of each convolutional layer before forwarding to the next convolutional layer. More details of the robust texture-embedded supplementary method are explained in the next subsection. The proposed architecture contains eight layers—a robust texture-embedded supplementary, three convolutional, one single energy, and three fully connected layers as shown in Fig. 8. The convolutional layer extracts feature map by using 3×3 convolutional filters with the stride of 2 and 1×1 padding. The ReLU non-linearity and normalization are applied to the feature map of every convolutional layer. An energy layer [17] is inserted before fully connected layer to extract energy descriptors from the feature maps of the last convolutional layer. The three fully connected layers are used for classification in the same way as AlexNet.

3.3 Robust Texture-Embedded Supplementary Method for CNN

From observation of texture patterns, their characteristics contain multiple scales with uncertain directions. Thus, robust texture becomes a key factor for supplementing the CNN architecture. In other words, the robust texture always contains multi-scale and multi-orientation contents. In order to meet requirement, a robust texture-embedded supplementary method is proposed to preserve important textures as much as possible and is embedded into the CNN architecture under a single pipeline. The proposed algorithm can be described as follows.

Step 1: Transform an input image ψ into the Fourier spectrum Ψ. The input image and its Fourier spectrum can be seen in Figs. 6(a) and 6(b), respectively.

Step 2: Segment Ψ by using 2D Littlewood-Paley EWT [24], [25] to obtain a set of subbands as shown in Fig. 6(c). It can be written in a vector form as Eq. (1).

$$\Psi = [\Psi_1, \Psi_2, \Psi_3, ..., \Psi_n] \tag{1}$$

where Ψ_i is the i^{th} Fourier spectrum subband and n^{th} is the last subband.

Step 3: Remove the first and the last subbands, Ψ_1 and Ψ_n, of Fourier spectrum Ψ as expressed in Eq. (2). The result of Ψ_T is shown in Fig. 6(f).

$$\Psi_T = \bigcup_{i=2}^{n-1} \Psi_i \tag{2}$$

Step 4: Apply inverse Fourier transform to Ψ_T to obtain texture component ψ_t as shown in Fig. 6(i).

Step 5: Extract multi-scale (MS) and multi-orientation (MO) features $\delta_{u,v}$ from a texture component ψ_t by using Gabor wavelet filter [26], [27] as defined by Eq. (3)

$$\delta_{u,v} = \psi_t * \phi_{u,v} \tag{3}$$

where * is the convolution operator, $\delta_{u,v}$, is the multi-scale and multi-orientation features. $\phi_{u,v}$ is the Gabor filter at scale u and orientation v. The multi-scale and multi-orientation features with two scales and eight orientations are illustrated in Figs. 7(b) and 7(c), respectively. Then, a set of multi-scale and multi-orientation features, δ, is defined by Eq. (4).

$$\delta = \{\delta_{u,v} \mid u \in \{1, ..., 5\}, v \in \{1, ..., 8\}\} \tag{4}$$

Step 6: Construct a set of robust-texture descriptors $\gamma_{l=1}$ at level l by concatenating ψ_t and δ.

As mentioned in the previous section, the deep convolutional layer makes the texture lost. It is a cause of the low recognition accuracy rate, since it cannot recognize (i) similarly shaped objects with different textures of im- ages often assigned into different classes [16] and (ii) different shaped objects with similar textures of images often assigned into the same class [16]. Therefore, texture compensation in each depth of convolutional layer is required. In order to complete this requirement, a set of robust-texture descriptors are constructed in the form of levels to embed into each convolutional layer. Robust-texture descriptor construction at level l+1 is described as follows.

Step 1: Reduce texture component size at level l from the texture resolution M-by-N to $\frac{M}{2}$ -by-$\frac{N}{2}$. The texture component at level l+1, ψ_{tl+1}, is obtained by using subsampling operation [28], [29] as shown in Fig. 8.

Step 2: Extract multi-scale and multi-orientation features δ_{l+1} from ψ_{tl+1} by using Eqs. (3) and (4), respectively.

Step 3: Construct a set of robust-texture descriptors γ_{l+1} by concatenating ψ_{tl+1} and δ_{l+1}.

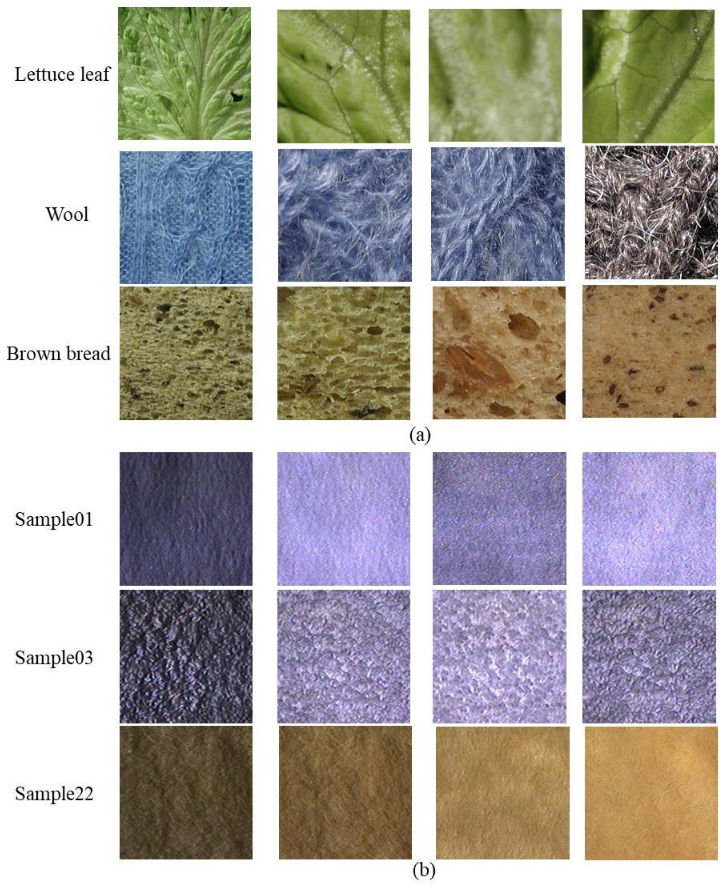

Fig. 9: Characteristics of the two challenging datasets: (a) Varied texture images within the same class of the kth-tips-2b dataset; each row shows images from four sets of three following classes—lettuce leaf, wool, and brown bread classes; (b) Examples of images in CUReT dataset.

The outcome of the robust texture-embedded supplementary method is a set of robust-texture descriptors generating for each convolutional layer as illustrated in Fig. 8.

4.0 DATASETS, EXPERIMENTS, AND RESULTS AND DISCUSSION

This section describes three publicly available datasets that are used in this experiment, experimental evaluation with these datasets, experimental results and discussion of the results.

4.1 Datasets Used in This Experiment

As mentioned in Section 1.0, texture analysis is a challenging problem due to the following reasons: (i) images in the same class may naturally contain many different viewpoints, different scales, uneven illuminations, etc.; (ii) similarly shaped objects with different kinds of texture are often assigned to the different class; and (iii) different shaped objects with same kinds of texture are often assigned to same classes. In order to evaluate the performance of the proposed method while covering all of these three challenges, three public texture datasets that reflect them are used.

To cover the first challenge, the performance of the proposed method is evaluated by using kth-tips-2b [15] and CUReT [30] datasets. The kth-tips-2b dataset contains 432 texture images of 11 classes. Each class consists of four sets and each set contains 108 images. Each set of images is used for training one at a time while the re-maining three sets are used for testing. The CUReT dataset contains 61 classes. Each class consists of 92 images. Each class is equally divided into two sets, a set of 46 images for training and a set of 46 images for testing. The

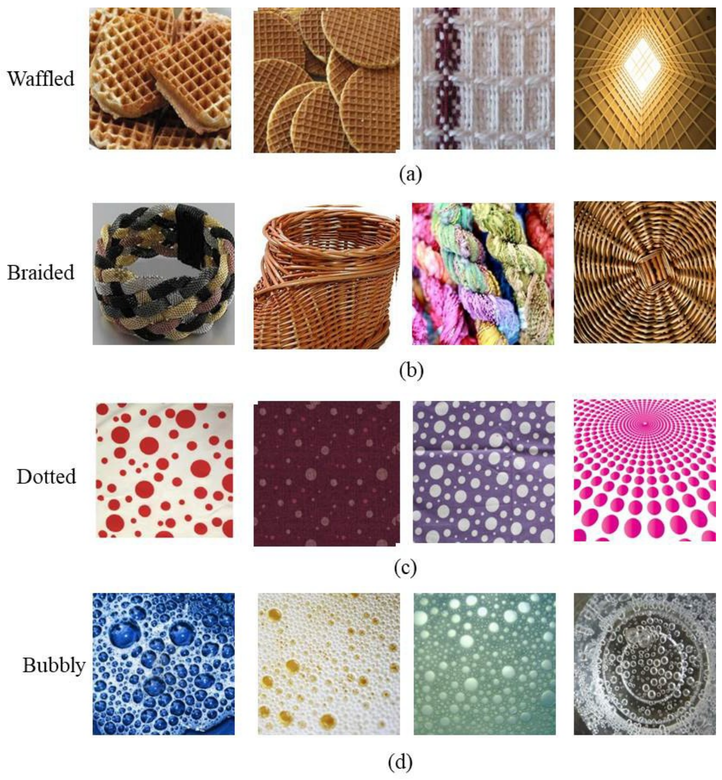

Fig. 10. An example of images in DTD dataset; each row shows an image from Waffled, Braided, Dotted and Bubbly classes; a single image contains both texture and object(s).

experiment is repeated 20 times. For the respective kth-tips-2b and the CUReT datasets, each image in the same class shows a different viewpoint, illumination and scale condition as shown in Figs. 9(a) and 9(b). To cover the second and the last challenges, a Describable texture dataset (DTD) [16] is used for evaluation of the performance of the proposed method. This dataset contains 47 classes of 120 images each. Forty images in each class are used for training, another 40 for validating, and the other 40 for testing. This dataset includes 10 splits. Most images in this DTD dataset contain both (an) object(s) of various shapes and different textures. Images in the same class often contains (an) object(s) with the different shape but the same texture such as a Waffled class or a Braided class as shown in Figs. 10(a) and 10(b), respectively. Images from different classes often contain (an) object(s) with the same shape but the different texture such as a Dotted class and a Bubbly class as shown in Figs. 10(c) and 10(d), respectively.

5.0 EXPERIMENTS

Ordinarily, any architectures of the convolutional neural network type will have a different depth of convolutional layer because each of them would be designed to work specifically with a different kind of dataset such as lung texture and forest species datasets. Therefore, the preliminary experiment is the proposed architecture with the depth of convolutional layer. This experiment aims to test the recognition accuracy rate of the proposed architecture based on three different kinds of datasets of which the depth of its convolutional layer is progressively increased. The proposed architecture is tested to four convolutional layers of depth.

The results in Table 1 show the average recognition accuracy rate and standard deviation on each dataset of each depth of convolutional layer. The best recognition accuracy rate achieves by our proposed architecture is used to compare with those achieves by the following baseline methods: AlexNet [7], T-CNN [17], and wavelet CNN [21]. The results in Table 2 show the average recognition accuracy rate and standard deviation on each dataset of all 4 architectures. All images in every dataset are resized to 227 ×227 for training. Our proposed architecture is trained from scratch by using a learning rate of 0.001 and a weight decay of 0.0005 [17].

Table 1: Comparison of the recognition accuracy rates that our proposed architecture achieved at 4 depth levels on kth-tips2-b, DTD, and CUReT datasets

Dataset	**Level 1**	**Level 2**	**Level 3**	**Level 4**
kth-tips2-b	55.0 ± 2.0	**63.1± 1.9**	60.1± 1.9	58.4 ±1.8
DTD	20.3 ±1.1	25.2± 1.3	**34.8± 1.2**	32.0± 1.2
CUReT	94.9 ±0.6	**98.1± 0.7**	97.2± 0.7	96.1± 0.8

Table 2: Comparison of the recognition accuracy rates achieved by the four architectures trained from scratch on kth-tips2-b, DTD, and CUReT datasets

Dataset	**AlexNet**	**T-CNN**	**Wavelet CNN**	**Proposed Method**
kth-tips2-b	47.6 ±1.4	48.7± 1.3	60.5±2.1	**63.1 ±1.9**
DTD	22.7 ±1.3	27.8± 1.2	32.3± 0.8	**34.8± 1.2**
CUReT	<u>98.7 ±0.6</u>	98.2± 0.6	98.0± 0.8	98.1± 0.7

5.1 Experimental Results and Discussion

The experimental results of the proposed architecture with the depth of convolutional layer are shown in Table 1. Our proposed architecture achieves the highest recognition accuracy rate on the kth-tips2-b and CUReT datasets at depth level 2, whereas it achieves the highest recognition accuracy rate on the DTD dataset at depth level 3. The comparative experimental results show that the proposed architecture outperforms all of the baseline methods on the kth-tips2-b and DTD datasets, while its performance on the CUReT dataset is virtually the same as those achieved by the baseline methods as shown in Table 2. In particular, on the kth-tips2-b and DTD datasets, the improvement in performance of the proposed method is very significant when compared to those of AlexNet, T-CNN, and wavelet CNN: 15.5%, 14.4%, and 2.6% better, and 12.1%, 7.0%, and 2.6% better, respectively. However, the wavelet CNN requires data augmentation for performance improvement whereas our architecture does not. On the CUReT dataset, the recognition accuracy rate of the proposed method is slightly different from those of the baseline methods, but the differences are minute. For instance, the difference in the recognition rate of the proposed method to that of Alexnet which achieves the highest recognition accuracy rate on this dataset is only 0.6%.

Figs. 11, 12, and 13 show examples of the images in the three datasets that are correctly recognized to be in the same class by all four methods. Each row shows examples of the images in the same class. For the images in the kth-tips2-b dataset in Fig. 11, it can be seen in these figures that our proposed method and wavelet CNN method are able to correctly recognized different-looking images (different viewpoints, different scales, uneven illuminations, etc.) to be of the same class while the AlexNet and T-CNN methods are able to correctly recognize only similar-looking images (slightly different viewpoints, slightly different scales, slightly uneven illuminations, etc.) to be of the same class. For the images in the DTD dataset in Fig. 12, it can be seen that AlexNet is able to correctly recognize shaped objects to be of the same class while T-CNN is able to correctly recognize mainly shaped objects but also a few textures, whereas wavelet CNN and our proposed methods are able to correctly recognize both shaped objects and textures as belonged to the same class equally well. On the other hand, the images in the CUReT dataset in Fig. 13 are all images of textures and in this case, all of the tested methods correctly recognized them as belonged to the same class equally well. The reason that our proposed architecture achieves better recognition accuracy rate is that it is able to resolve the texture loss by texture compensation and to provide the significant textures by texture supplement as schematically shown in Fig. 8. Figs. 14(b) and 14(c) show an example of feature maps from the last convolutional layer of AlexNet and one from the proposed method, respectively. The black pixels in each feature map signified a loss of contents. The red border circumscribes the area of one feature. When the contents in the feature map processed by our proposed architecture and the contents of the same feature map processed by the AlexNet architecture are compared as can be observed in Figs. 14(a) and 14(b), it can be seen that the proposed method is able to preserve the contents whereas the AlexNet architecture suffers a loss of contents. This leads to the improvement in the recognition accuracy rate.

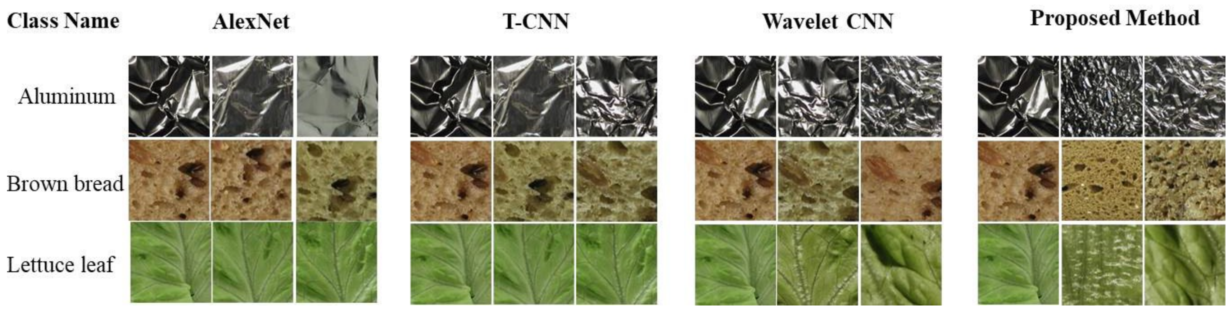

Fig. 11: Examples of images correctly recognized in the same class by AlexNet, T-CNN, wavelet CNN, and our proposed architecture in kth-tips2-b dataset, each row shows the image from Aluminum, Brown bread, and Lettuce leaf classes.

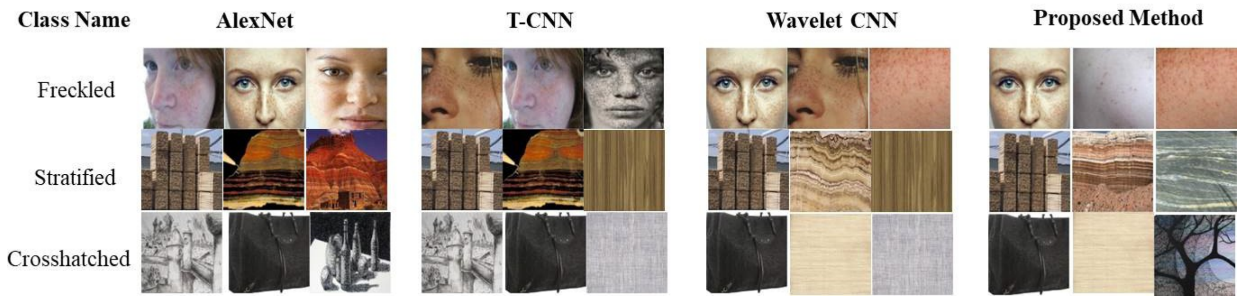

Fig. 12: Examples of images correctly recognized in the same class by AlexNet, T-CNN, wavelet CNN, and our proposed architecture in DTD dataset, each row shows the image from Freckled, Stratified, and Crosshatched classes.

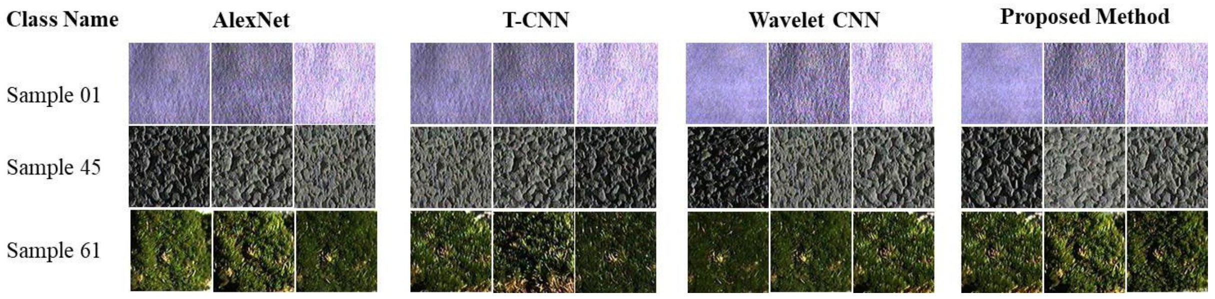

Fig. 13: Examples of images correctly recognized in the same class by AlexNet, T-CNN, wavelet CNN, and our proposed architecture in CUReT dataset, each row shows the image from Sample 01, Sample 45, and Sample 61 classes.

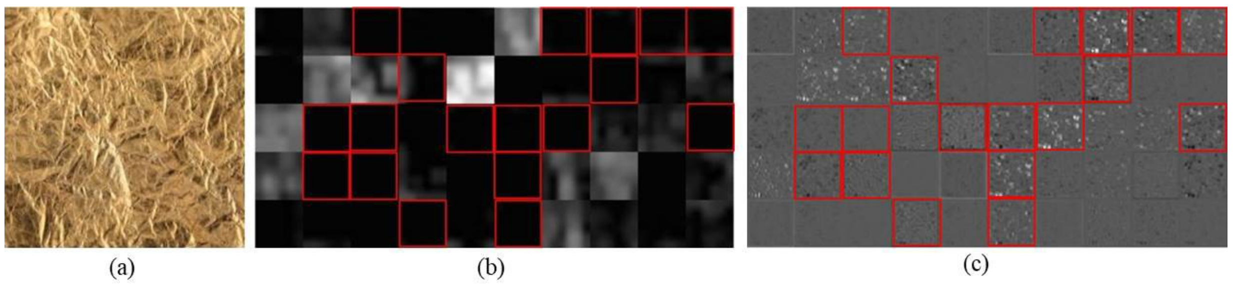

Fig. 14: Comparison of a loss of contents: (a) Input image; (b) feature map of the last convolutional layer from the input image achieved by AlexNet; (c) feature map of the last convolutional layer from the input image achieved by our proposed architecture.

6.0 CONCLUSION

In this paper, we have proposed a modified CNN architecture called a robust-texture convolutional neural network (RT-CNN). This prominent architecture is that it can work well with both complex shape and texture classification tasks. The main contributions of the proposed architecture are (i) compensation of lost textures with texture components adaptively decomposed by 2D Littlewood-Paley empirical wavelet transform (2D Littlewood-Paley EWT) and (ii) supplement of significant textures with multi-scale and multi-orientation features efficiently extracted by Gabor wavelet. Both the compensation of lost textures and the supplement of significant textures are embedded to the conventional CNN architecture by placing those texture feature images in the front of the first convolutional layer and concatenating to each convolutional layer. When the implementation of the proposed architecture is tested with two challenging texture datasets, our proposed RT-CNN can significantly improve the recognition accuracy rate when compared to all test baselines. On the typical dataset, the proposed method is still better than two baseline methods: T-CNN and Wavelet-CNN, but is slightly worse, 0.6%, than AlexNet. This achievement proves that the proposed architecture really works well with both complex shape and texture classification tasks. In future work, the RT-CNN's capability will be extended to a fully adaptive RT-CNN; that is, the number of scale and orientation textures will be based on the behavior of real image textures.

REFERENCES

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-Based Learning Applied to Document Recognition, in Proceedings of IEEE, 1998, pp. 2278-2324.

[2] H. Khalajzadeh, M. Mansouri, and M. Teshnehlab, Hierarchical Structure Based Convolutional Neural Network for Face Recognition, International Journal Computational Intelligence and Applications, Vol. 12, No. 3, 2013.

[3] C. F. Tivive and A. Bouzerdoum, A hierarchical Learning Network for Face Detection with in-Plane Rotation, Journal of Neurocompution Vol. 71, No. 16, 2008.

[4] A. Razvan-Daniel, Human Face Recognition Using Convolutional Neural Netwoks, Journal of Electrical and Electron Engineering, October 2009.

[5] Z. Zhao, S. Yang, and X. Ma, Chinese License Plate Recognition Using a Convolutional Neural Network, in Proceedings of IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, Vol. 1, 2008, pp. 27-30.

[6] I. Paliy, F. Lamonaca, V. Turchenko, D. Grimaldi, and A. Sachenko, Detection of Micro Nucleus in Human Lymphocytes Altered by Gaussian Noise Using Convolution Neural Network, in Proceedings of IEEE International Instrumentation and Measurement Technology Conference, 2011, pp. 1-6.

[7] A. Krizhevsy, L. Sutskever, and E. G. Hinton, Imagenet Classification with Deep Convolutional Neural Net- works, in Advances in Neural Information Processing Systems, 2012, pp. 1097-1105.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, and S. Reed, Going Deeper with Convolutions, in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1-9.

[9] K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, International Conference on Learning Representations (ICLR), 2015.

[10] K. He, X. Zhang, S. Ren, and J. Sun, Deep Residual Learning for Image Recognition, in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778.

[11] R.Girshick, J. Donahue, T. Darrell, and J. Malik, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[12] J. Long, E. Shelhamer, and T. Darrell, Fully Convolutional Networks for Semantic Segmentation, Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3431-3440.

[13] R. Girshick, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, arXiv:1506.01497[cs.CV], 2015.

[14] S. Ren, K. He, R. Girshick, and J. Sun, Fast R-CNN, in Proceedings of IEEE International Conference on Computer Vision (ICCV), 2018, pp. 1440-1448.

[15] E. Hayman, B. caputo, Ma Fritz, and J. O. Eklundh, On the Significance of Real World Conditions for Material Classification, in Computer Vision-ECCV, Springer, 2004, pp. 253-266.

[16] M. Cimpoi, S. Maji, L. Kokkinos, S.Mohamed, and A. Vedaldi, Describing Textures in the Wild, in Computer vision and pattern recognition (CVPR), IEEE Conference, 2014, pp. 3606-3613.

[17] V. Andrearczyk and P. F. Whelan, Using Filter Banks in Convolutional Neural Networks for Texture Classi- fication, Pattern Recognition Letters, 2016, pp. 63-69.

[18] G. L. Hafemann, S. L. Oliveira, and P. Cavalin, Forest Species Recognition Using Deep Convolutional Neural Networks, in Proceedings of International Conference on Pattern Recognition, 2014, pp. 1103-1107.

[19] E. Park, W. Kim, Q. Li, J. Kim, and H. Kim, Fingerprint Liveness Detection Using CNN Feature of Random Sample Patches, in Proceedings of International Conference of the Biometrics Special Interest Group (BIOSIG), 2016, pp. 1-4.

[20] Q. Wang, Y. Zheng, G. Yang, W. Jin, X. Chen, and Y. Yin, Multi-Scale Rotation-Invariant Convolutional Neural Networks for Lung Texture Classification, IEEE Journal of Biomedical and Health Informatics, 21 May 2016.

[21] S. Fujieda, K. Takayama, and T. Hachisuka, Wavelet Convolutional Neural Networks for Texture Classifica- tion, arXiv:1707.07394 [cs.CV], 24 July 2017.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research, 2014, pp. 1929-1958.

[23] A. Akansu and P. Haddad, Multiresolution Signal Decomposition, San Diego: Academic Press. ISBN 9780080512297, 13 October 2000.

[24] J. Gilles, G. Tran, and S. Osher, 2D Empirical Transforms. Wavelets, Ridgelets, and Curvelets Revisited, SIAM Journal on Imaging Sciences, Vol. 7, 1 January 2014.

[25] J. Gilles and K. Heal, A Parameterless Scale-Space Approach to Find Meaningful Modes in Histograms Application to Image and Spectrum Segmentation, arXiv: 1401.2686v1 [cs.CV], 13 January 2014.

[26] W. Nunsong and K. Woraratpanya, Modified Differential Box-Counting Method Using Weighted Triangle- Box Partition, in Proceedings of International Conference on Information Technology and Electrical Engi- neering (ICITEE), 2015, pp. 221-226.

[27] W. Nunsong and K. Woraratpanya, An Improved Finger-Knuckle-Print Recognition Using Fractal Dimen- sion Based on Gabor Wavelet, in Proceedings of International Joint Conference on Computer Science and Software Engineering (JCSSE), 2016, pp. 1-5.

[28] W. T. Freeman and E. H. Adelson, The Design and Use of Steerable Filters, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 13, No. 9, 9 September 1991.

[29] H. Mahersia and K. Hamrouni, Using Multiple Steerable Filters and Bayesian Regularization for Facial Ex- pression Recognition, Journal of Engineering Applications of Artificial Intelligence, Vol. 38, pp. 190-202.

[30] K. J. Dana et al., Reflectance and Texture of Real-World Surfaces, ACM Trans. Graph (TOG) 18 (1), 1999, pp. 1-34.