10 Convolutional Neural Network for Classification of Skin Cancer Images

Giang Son Tran, Quoc Viet Kieu, and Thi Phuong Nghiem
ICTLab, University of Science and Technology of Hanoi, Vietnam Academy of Science and Technology, Hanoi, Vietnam

CONTENTS

10.1 INTRODUCTION

Skin cancer is a severe and popular cancer disease. It is the abnormal development of skill cells in the outermost layer of skin. If not early diagnosed and treated, these cells will multiply uncontrollably and become malignant tumors [1]. Skin cancer usually appears on sun-exposed parts of the skin such as the face, neck, or hands. In

DOI: 10.1201/9781003217497-10

these parts, the sun's unhealthy ultraviolet (UV) rays are absorbed into the skin and gradually cause skin damage [1]. Two general categories of skin cancer include (1) melanoma skin cancer and (2) nonmelanoma skin cancer. In the first case, melanoma, the deadliest type of skin cancer, presents a malignant skin lesion arising from melanocytes which are skin cells producing melanin pigment. In the second case, nonmelanoma skin cancer is usually diagnosed by clinical images and histopathology images such as cut lesions in rhombohedral or elliptical shapes by surgical methods; burning laser; or radioactivity, radiotherapy (X-rays, radium rays) [2].

It was reported by [2] that each day there are nearly 10,000 people having skin cancer in the United States, and two people die every hour. Moreover, the treatment cost of skin cancer is very high. For example, in the United States, around $8.1 billion is used each year for skin cancer treatment [2]. In 2021, more than 5,400 people worldwide are estimated to get death due to nonmelanoma skin cancer each month, in which around 7,180 people (64% men, 36% women) will die of melanoma [2]. With large numbers of deaths caused by skin cancer, early detection will save lives for many patients. The statistics show that if a melanoma patient is treated appropriately at an early stage, they can achieve up to a 99% 5-year survival rate [2]. Therefore, a patient's survival rate can be enhanced by early detection, and diagnosis of skin cancer can help enhance the survival rate of patients.

Dermoscopy images are widely used by dermatologists to examine suspicious skin lesions [3]. To do so, the doctor usually looks at the dermoscopic images of skin lesions to determine if there exists cancer and, if so, which type of cancer the lesion may represent. Nevertheless, one problem that arises is that to analyze these million images of skin cancer is a massive workload for dermatologists due to the high variance of size, shape, texture, location between the healthy skin and the damaged skin.

Thanks to advances in machine learning technologies, computers are used to automate the detection process of malignant skin lesions [4]. Using computer methods to classify skin cancer, we can ease the workload of dermatologists during the clinical stages [5–7]. As a result, the time and treatment cost of skin cancer can be reduced. The general pipeline for skin lesion detection includes data preprocessing, feature extraction, image classification, and disease diagnosis. The classification of skin lesion images is essential to support the subsequent step of disease diagnosis in the general pipeline.

In this chapter, the main goal is to develop an automatic system to aid dermatologists in skin cancer examination and diagnosis. Specifically, we propose an automatic system using deep learning techniques to categorize multiple classes of skin lesions. Our system utilizes data preprocessing and augmentation to enlarge data on skin cancer. After that, transfer learning is applied to fine-tune a pre-trained model for extracting and classifying the image features. The experiments conducted on the publicly available ISIC 2019 dataset are used to measure the effectiveness of our proposed system for skin lesion image classification.

10.2 STATE-OF-THE-ART

Traditional approaches usually use image processing techniques and conventional learning models to classify skin lesion images. For example, Murugan et al. [8] utilized the support vector machine (SVM) as a classifier to determine skin lesion images as benign or melanoma. With different experiments, the experiments show that SVM produced better classification results compared to several other methods. Farooq et al. [9] segmented cancerous areas in the skin lesions by combining active contours and watershed techniques. Later, SVM was used to classify the cancer moles. Finally, an additional classification artificial neural network is used to fine-tune the SVM results and check the indeterminate cases produced from SVM's output. However, these traditional methods highly depend on preprocessing and post-processing steps of images and the hand-crafted features of skin lesion images. These methods are difficult to achieve satisfactory results due to the high variability of size, shape, and texture between healthy skin and damaged skin.

Recently, deep neural networks have been popular in skin lesion image classification. Following this direction, one common approach is the use of transfer learning to fine-tune pre-trained models for this task. For example, Chaturvedi et al. [10] proposed an automated system in order to categorize multiple classes of skin cancer. The authors performed fine-tuning of deep learning networks to improve classification performance by (1) adding "ReLU" activation to dense layers, (2) putting dropout and softmax layers at the end of the network, and (3) adapting values of hyperparameters. The maximum performance for an individual network is 93.20% of accuracy, and for the ensemble model is 92.83%. Through extensive experiments, ResNeXt101 is recommended by the authors for skin cancer image's multi-class classification.

Dorj et al. [11] utilized a pre-trained deep network called AlexNet to extract features of skin cancer images from four types, including SCC, BCC, MEL, and actinic keratosis (AK). SVM classifier is then applied to classify the skin cancer images. From the experiments, the method obtains maximum performances for accuracy, sensitivity, and specificity are 95.1% for SCC, 98.9% for AK, and 94.17% for SCC, respectively. Similarly, the minimum performances are 91.8% accuracy for BCC, 96.9% sensitivity for SCC, and 90.74% specificity for MEL. It is noticed that this work gains very good sensitivity results for AK and SCC cancers.

Hosny et al. [12] fine-tuned a pre-trained deep learning model, namely AlexNet, to improve the classification performance of skin lesions. Specifically, the authors replaced the pre-trained AlexNet's last layer with a softmax activation function to categorize three types of skin lesions (MEL, common nevus, and atypical nevus). The method was evaluated on the PH2 public dataset and achieved 98.61% accuracy, 98.33% sensitivity, 98.93% specificity, and 97.73% precision. The results of this work are much higher than the existing methods. However, the trained and tested dataset was a small one compared to other datasets such as ISIC challenge datasets.

Although these methods have obtained satisfactory results, the classification performance of skin cancer in dermoscopic images still needs further improvement to meet the high variability in size, shape, texture, and location of skin lesions. This

chapter aims to improve the skin lesions' multi-class classification by employing and fine-tuning pre-trained Xception, a deep learning network. A detailed description of the materials and methods to implement this objective is presented in the next section.

10.3 MATERIALS AND METHODS

We propose an automatic system using a deep neural network and transfer learning to classify multiple types of skin cancer images. Figure 10.1 demonstrates a visualization of our proposed method. From the figure, we perform data preprocessing to clean input data and data augmentation to enhance training samples. Some parts of the pre-trained Xception model are kept intact for extracting features of input images, while the remaining parts are fine-tuned for fitting with the problem of multi-class classification of skin lesions.

10.3.1 Data Preprocessing and Augmentation

An international organization named International Skin Imaging Collaboration (ISIC) aims to support researchers worldwide about skin cancer diagnosis. Through its challenges in 2017, 2018, 2019, and 2020, ISIC provides the largest publicly available datasets of skin lesions' dermoscopy images, which now become the benchmark datasets for scientists worldwide. All photos in ISIC datasets were collected from various skin microscopes, from different surgical sites and several hospitals.

In this chapter, we use the ISIC 2019 challenge dataset [13–15] to train and test our model. The dataset contains 25,331 images collected from three other datasets, namely HAM10000, BCN_20000, and MSK. The HAM10000 dataset contains 12,413 images of size 600x450 centered and cropped around the lesions, while BCN_20000 has 10,015 images of size 1024x1024. The pictures of BCN_20000 dataset is challenging since they are uncropped and skin lesions are in different locations, scale, and angle to spot out. The last one is MSK with 819 images of various sizes. Qualified dermatologists created the ground truth labels.

Table 10.1 demonstrates the detailed description of each class with the corresponding number of images, and Figure 10.2 shows image examples in ISIC 2019 challenge dataset. From the tables, it can be seen that ISIC 2019 challenge dataset is imbalanced. There are only 239 images in the smallest class (Dermatofibroma class) and up to 12,875 photos in the dataset's largest class (Melanocytic nevi class). The properties of skin lesion images in the ISIC 2019 dataset vary differently from irregular boundaries, thick hair, or dark edges.

Having 25,331 images from ISIC 2019 challenge dataset, we randomly get 2,534 images (10%) to form the test set. For the remaining 22,797 images (90%), since there exist cases where the same lesion of one person may have several images, we removed 15,959 duplicate images to clean data. After this step, we obtained 5,357 images without duplication. From this number of data, we get 1,072 photos (20%) for the validation set. The last 4,285 images (80%) were merged with 15,959 photos to create the training set of 20,244 images.

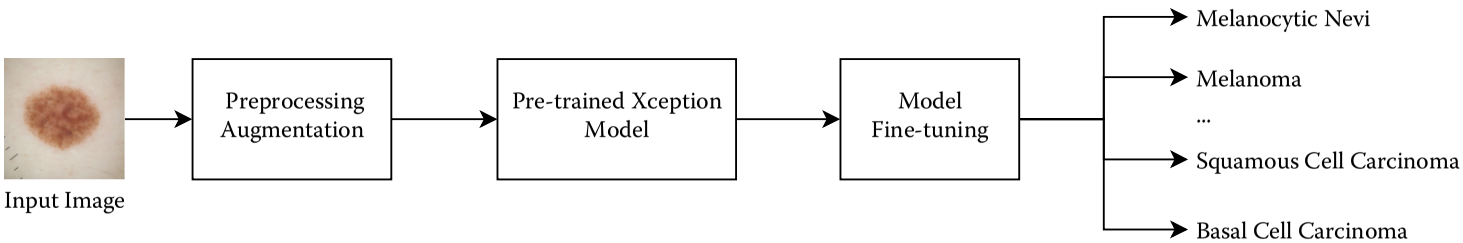

FIGURE 10.1 Our classification system of skin lesion images.

TABLE 10.1
Description of ISIC 2019 Challenge Dataset [13–15]

Class	Number of Images	Ratio (%)
Melanocytic nevi (NV)	12,875	50.83
Melanoma (MEL)	4,522	17.85
Basal Cell Carcinoma (BCC)	3,323	13.12
Benign keratosis-like lesions (BKL)	2,624	10.36
Actinic keratoses (AK)	867	3.42
Squamous Cell Carcinoma (SCC)	628	2.248
Vascular lesions (VASC)	253	1.00
Dermatofibroma (DF)	239	0.94

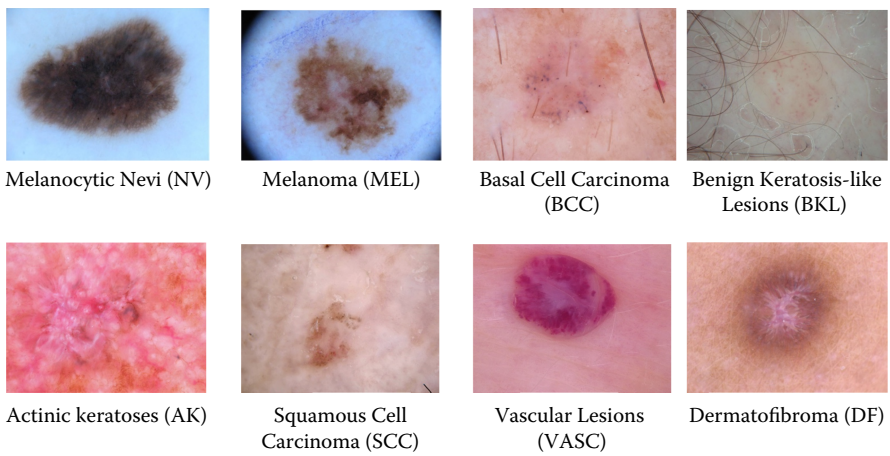

FIGURE 10.2 Examples of ISIC 2019s skin lesion images provided by the dataset [13–15].

For data preprocessing, it is noticed that input images from the ISIC 2019 challenge dataset have inconsistent dimensions with various sizes, while our classification model requires fixed input size. Due to this, image scaling is necessary to feed our input image data to train the classification model. Therefore, we scaled all images to the size of 299 × 299 pixels to match with the Xception model's input size.

10.3.2 Data Augmentation

As can be seen from Table 10.1, the distribution of skin lesion classes in the ISIC 2019 challenge is very imbalanced. This problem can make the model learn bias towards the major classes more than the minor classes. To avoid this problem, we

TABLE 10.2
Training Images After Augmentation

Class	Training Images After Augmentation	Ratio (%)
Melanocytic nevi (NV)	9128	13%
Melanoma (MEL)	9104	13%
Basal Cell Carcinoma (BCC)	9098	13%
Benign keratosis-like lesions (BKL)	9114	13%
Actinic keratoses (AK)	8960	13%
Squamous Cell Carcinoma (SCC)	7818	11%
Vascular lesions (VASC)	8358	12%
Dermatofibroma (DF)	8506	12%

apply different data augmentation techniques to increase more image samples from minor classes in order to be equal to the major classes. Another benefit of data augmentation is to have more samples for the training set, which is often required for deep learning models.

Having 20,244 images from ISIC 2019 challenge dataset, the following data augmentation methods are utilized:

- Random rotation from 0° to 180°
- Shift horizontally and vertically: 10%
- Zoom in the inputs: 10%
- Flip horizontally and vertically

After augmentation, we have 70,086 images for the training set, in which the ratio of images in each class is much more balanced compared to the original dataset. Table 10.2 shows a detailed description of images in each category in our training set.

Table 10.3 presents a description of our training and testing dataset for the experiments in this work. It is noticed that the test set contains only original images from the ISIC 2019 dataset, while the validation set consists of clean data with duplicate image removal and the training set contains both original and augmented image data.

10.3.3 Classification Models

10.3.3.1 Convolutional Neural Network (CNN)

CNN is an important building block of many intelligent systems with high accuracy [16]. Generally, a CNN consists of an input layer, several hidden layers for feature extraction, an optionally fully connected layer for classification, and an output layer (Figure 10.3). A CNN hidden layer is usually a convolutional layer followed by an activation function. Additionally, after the activation function, a pooling layer can

TABLE 10.3
Detail of Training and Testing Data in Our Experiments

	Training Set	Validation Set	Test Set
Number of images	70,086	1,072	2,534

be added in order to down-sample the input features but does not affect the image channels. The fundamental purpose of a CNN hidden layer is to extract spatial information of the image features automatically. A CNN network can contain as many as needed hidden layers for better accuracy performance. After the feature extraction step, a dense layer can be used to perform the classification of image features. This dense layer is usually followed by an activation function for generating probabilistic distribution of output values.

The **convolutional layer** is the most important hidden layer in a CNN. It applies convolutional products with many filters/kernels on the input image, then is followed by an activation function to break the linearity of the CNN network. It is the heavy part of a CNN that performs most computational tasks to learn the meaningful features. The output of a feature map j at the convolutional layer l is defined as follows [10]:

$$C_j^l = f\left(\sum_{i=1}^{N^{l-1}} C_i^{l-1} * w_{ij}^l + b_j^i\right) \tag{10.1}$$

Where l is the layer number, C_j^l is the output of activation function of feature map j, f is the activation function, w_{ij}^l are kernel weights from the feature map i at the previous layer to the feature map j at the current layer, N^{l-1} is the feature map number of the previous layer and b_j^i is the bias coefficient between the two feature maps i and j.

An **activation function** transforms a neuron's output from linear to non-linear. In our model, after each convolutional layer, we use ReLU activation function. At the end of the network, after the fully connected layer, we apply softmax activation function. ReLU is defined as follows:

$$relu(x) = \begin{cases} x & for \ \ x \geq 0, \\ 0 & for \ \ x < 0. \end{cases} \tag{10.2}$$

Where x is the vector output of convolutional layer before the activation function. $relu(x)$ is the ReLU activation function applied to x to transform its values from linear to non-linear. Softmax is defined as follows:

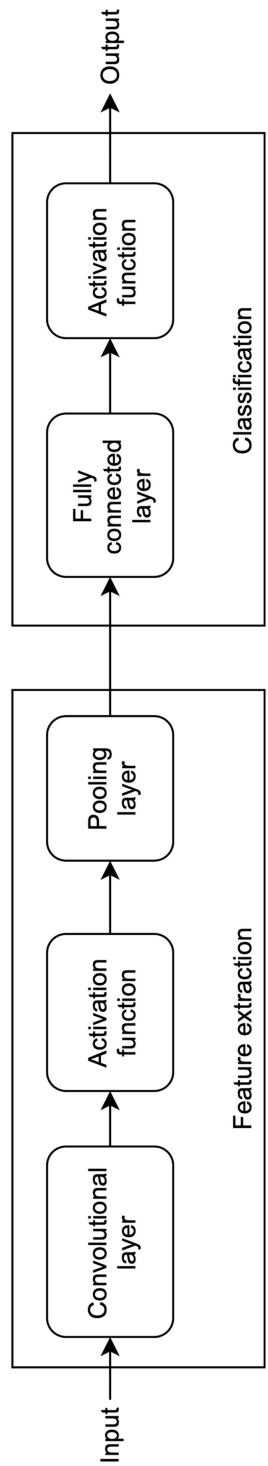

FIGURE 10.3 General architecture of a CNN.

$$softmax\,(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{10.3}$$

Where x is the input vector (the same as the fully connected layer's output) and K is the desired number of classes. The softmax activation function converts the output values by dividing each value by the total weight. The output is a vector of the predicted probabilities of the classes. The softmax activation function is beneficial in multi-class classification problems since it outputs the probability of each image over the K classes.

We use the categorical cross-entropy loss function to measure the difference between the model's output and the expected output. The definition of the loss function E is as follows [10]:

$$E = -ln\left(\frac{e^{\partial_p}}{\sum_i^K e^{\partial_i}}\right) \tag{10.4}$$

Where i is the iteration index, K is the number of classes, ∂_p is the network score for the positive class, and.

To optimize the loss function E, Adam optimizer [17] is applied. Specifically, after each iteration, weight and bias parameters of the network are updated as follows [10]:

$$\partial_{i+1} = \partial_i - \frac{\alpha \nabla \mathrm{E}(\partial_i)}{\sqrt{\nu_i} + \varepsilon} \tag{10.5}$$

Where α is the learning rate, ε is a minimal number to avoid zero in the denominator, ν_i is calculated using the following equation [10] with γ is the decay rate:

$$\nu_i = \gamma \nu_{i-1} + (1 - \gamma)[\nabla \mathrm{E}(\partial_i)]^2 \tag{10.6}$$

A **pooling player** performs down-sampling of input features without impacting the channel number of the image. It is usually inserted between convolutional layers to reduce the spatial dimensions. Consequently, it helps to decrease the number of training parameters for the next convolutional layer while keeping the depth of a CNN model unchanged. Average pooling layer or max-pooling layer are usually used in a CNN. The former calculates the average (mean) value of all elements in the filter, while the latter takes the maximum value from all filter elements. Nowadays, max pooling is widely used in CNN models for performing down-sampling input features between the convolutional layers.

A **fully connected layer** has all of its neurons connected to all neurons in the previous layer. This layer can also be called a dense layer. After applying hidden layers for feature extraction, a couple of dense layers are usually added to perform feature classification. One layer gathers the feature layers, and another converts the data from 3-D or 2-D into 1-D, representing probability output for each class.

10.3.3.2 Transfer Learning and Pre-trained Models

Following the rapid development of deep learning models, various well-known classification models are pre-trained on different datasets, for instance, CIFAR-100 or ImageNet [18]. The model's source codes and weights are usually publicized to the public. We call such models *pre-trained models*. A new model reusing some (or all) parts of a pre-trained one to another task is called a *transferred model*.

Utilizing and adapting a pre-trained model for new tasks is called *transfer learning* [19]. This progress is becoming popular since training a new deep learning model is very costly and time-consuming since it requires a vast number of data samples and parameters to train. Using transfer learning, we can get a better model based on the modifications of pre-trained models without training it from scratch.

The main steps of using transfer learning in this work are as follows:

1. Pre-trained model selection: find pre-trained models (those trained on a large and diverse image dataset) such that the pre-learned features can fit with skin lesion ones. After doing a survey, we decided to choose a modern pre-trained classification model provided by Google called Xception for this work.
2. Fine-tune pre-trained model: the next step is to analyze the model architecture to perform its fine-tuning. We need to decide which part of the pre-trained Xception will be kept and which part will be re-trained to learn new features from a new dataset of skin lesions.

Using the pre-trained model is essential for researchers to follow in their predecessor's accomplishments, leveraging existing pre-trained models to create new models for more specific target tasks and more practical applications.

10.3.3.3 Pre-trained Xception Model

Proposed by Chollet, 2017 [20], Xception is an improved version of Inception, a very deep popular CNN model produced by Google for image classification. The name "Xception" is short for "Extreme Inception". The principle of Xception architecture is the use of depthwise separable convolution layers. In such an architecture, instead of using the regular convolution layer, the author divided it into two separated types: depthwise and pointwise convolutions. In the former layer, each channel is convolved with a filter of depth 1. In the latter layer, the output feature maps of the depthwise convolution are convolved with 1x1 filters, which have the depth representing the desired number of channels in the output image. Figure 10.4 illustrates the general architecture of Xception with three blocks, namely entry, middle, and exit flows.

The entry flow takes input tensors of size $299 \times 299 \times 3$ and performs two blocks of convolutional layers and ReLU. After that, it performs various depthwise separable convolutional layers, followed by 3×3 max-pooling layers. There are skip connections with 1×1 convolution where the "add" operator is used to merge two tensors. The shape of input and output in each flow is also presented. For example,

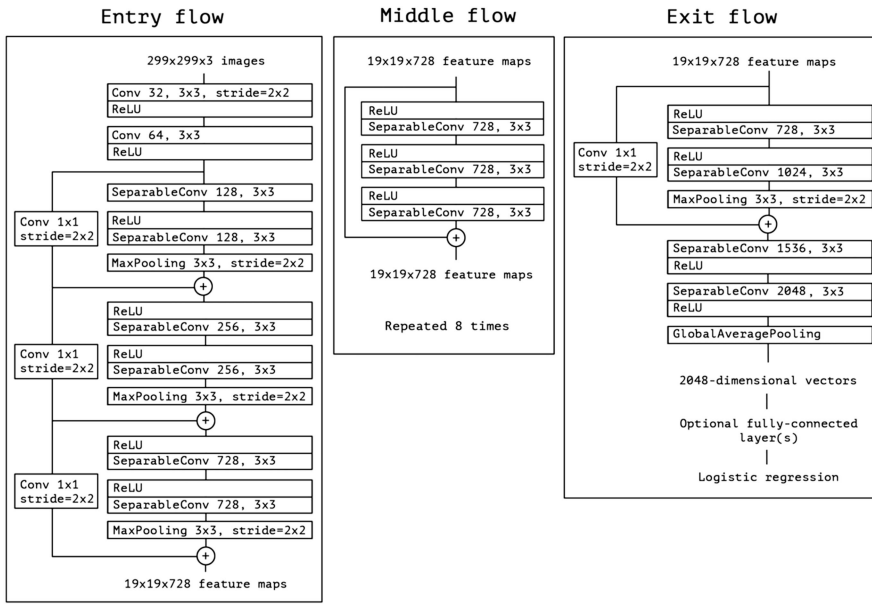

FIGURE 10.4 General architecture of Xception [20].

the dimension of the entry flow is $299 \times 299 \times 3$, and the output size of feature maps is $19 \times 19 \times 728$.

The middle flow takes feature maps of size $19 \times 19 \times 728$ as input and performs eight modules of depthwise separable convolution layers, one after the other. All of these modules utilize a stride of 1 without any pooling layers. As a result, the spatial size of the tensor stays the same as passed from the entry flow's output. Besides, the channel depth is the same as the input's depth of 728 since the middle flow modules have 728 filters.

The exit flow contains two convolution modules. The first module uses max pooling and the skip connection, while the second one uses global average pooling without skip connection. Besides, fully connected layers can be optionally used before passing the output to the logistic regression layer.

In this work, we employ the pre-trained Xception model. This model has been previously pre-trained on the popular ImageNet dataset [21]. Xception is currently one of the latest and most accurate models for classification problems provided by Google. The network has a total of 71 layers with 22.9 million parameters.

10.3.3.4 Xception Model Fine-tuning

In order to make the pre-trained Xception model work efficiently with the problem of multiple classifications of skin cancer, we perform the model fine-tuning as follows:

1. The batch size for model training and validation is set to 20.
2. The last two layers of the pre-trained Xception model are removed.

3. We append a dense layer to Xception for 8-class classification.
4. Dropout ratio is set to 0.25 to prevent overfitting.
5. We use Adam optimizer, having a learning rate of 10^{-2}.
6. Replace Logistic Regression with softmax.
7. Categorical cross-entropy loss function is used to measure the model performance.
8. The weights of the first 36 pre-trained layers are reused for extracting features, while the last 35 layers are re-trained to learn new features and perform class prediction of skin lesion images. We choose to reuse 36 layers of the pre-trained Xception model since these layers represent basic features, for instance, lines or edges, that can be suitable for skin lesion properties.
9. The model is trained using a total of 30 epochs.

To facilitate the experiments, we use ModelCheckpoint callback to save checkpoints of the model during training. The learning rate is halved after two epochs when the metric has stopped improving.

To perform Xception model fine-tuning, we used Keras as the deep learning framework. Keras contains different pre-trained models on the ImageNet dataset. Additionally, it supports various deep learning libraries as backend, for instance, TensorFlow or Theano. We performed our experiments using Keras 2.2.1 with TensorFlow 1.14 on Python 3.7.3, running on Debian 10.

Regarding hardware infrastructure, our experiments are conducted on a server equipped with a GeForce RTX 2080 Ti GPU.

10.3.3.5 Evaluation Metrics

After training the model, it is crucial to measure its effectiveness to categorize skin lesions into multiple classes. A confusion matrix is used to show a model's classification performance in each class. In a confusion matrix, one dimension presents the values of each actual class, while another dimension provides the predicted values of each class. A cell value shows the number of instances in which the model guesses correctly or incorrectly to a given actual class. As a result, a confusion matrix is generally used to evaluate a given classification model's performance.

A binary classifier has only positive and negative values. Prediction output can be one of these four cases:

- True positive (TP): a positive case is correctly predicted as positive.
- True negative (TN): a negative case is correctly predicted as negative.
- False positive (FP): a negative case is incorrectly predicted as positive (Type I error).
- False negative (FN): a positive case is incorrectly predicted as negative (Type II error).

For a multi-class classifier, the number of classes K is not 2, and there are neither positive nor negative cases. However, it can be desirable to measure the effectiveness of the classifier on each class. To do this, for each class $C_i, i = 1 \ldots \mathrm{K}$, we consider a binary classification in the test set: C_i as positive class while the rest

$\{C_j,\ j \neq i\}$ as negative class. As such, for each class C_i, we can calculate TP_i, TN_i, FP_i, and FN_i values based on the prediction output.

To measure the validity of a model, a set of more statistical metrics are used:

- **Accuracy** measures overall correct predictions:

$$Accuracy = \frac{\sum_0^{n-1}(TP_i + TN_i)}{\sum_0^{n-1}(TP_i + TN_i + FP_i + FN_i)}$$

- **Precision** measures the proportion of true positives among those classified as positives:

$$Precision = \frac{\sum_0^{n-1} TP_i}{\sum_0^{n-1}(TP_i + FP_i)}$$

- **Recall**, also called sensitivity, measures the proportion of predicted positives that are truly positive:

$$Recall = \frac{\sum_0^{n-1} TP_i}{\sum_0^{n-1}(TP_i + FN_i)}$$

- **Specificity** measures the proportion of predicted negatives that are truly negative:

$$Recall = \frac{\sum_0^{n-1} TN_i}{\sum_0^{n-1}(TN_i + FP_i)}$$

- **F1-score** is the precision's and recall's harmonic mean:

$$F1 = 2*\frac{Precision*Recall}{Precision + Recall}$$

F1-score is valid between half intervals (0,1]. F1-score indicates the quality of the classifier. In the best scenario, $F1 = 1$ when recall and precision are equal to 1.

- **Balanced Multi-class Accuracy (BMA)** is the mean recall of the confusion matrix:

$$BMA = \frac{1}{8}* \sum_{j=1}^{8} Recall_j$$

According to the ISIC 2019 challenge definition, BMA represents the diagnosis category score of the model. Hence, it is used to rank the participants of the challenge.

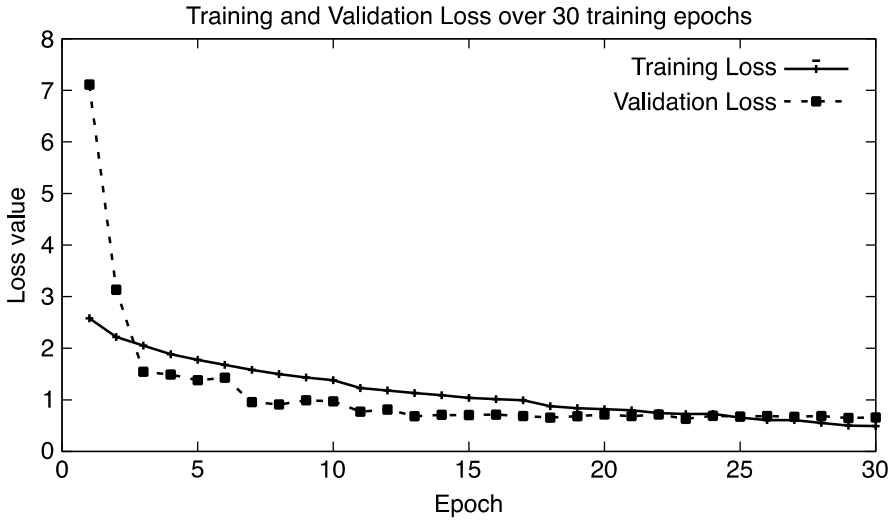

FIGURE 10.5 Learning curve of fine-tuned Xception model: training and validation loss.

10.4 EXPERIMENTAL RESULTS

10.4.1 Learning Performance

Figure 10.5 and Figure 10.6 illustrate the learning performance of the fine-tuned Xception model during training. From Figure 10.5, it can be observed that the validation loss is very close to the training loss, leading to a good generalization ability. Specifically, from epoch 25 onward, our model converges since the

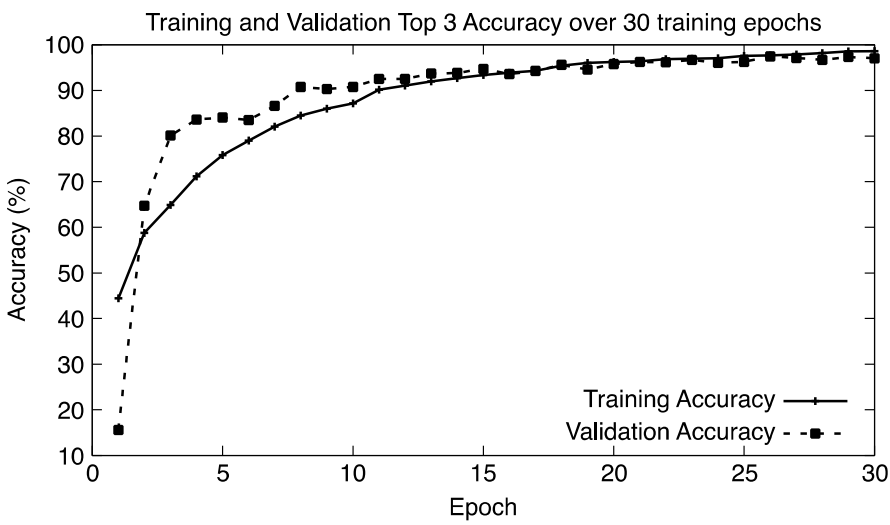

FIGURE 10.6 Learning curve of fine-tuned Xception model: training and validation accuracy.

difference between the two curves is negligible. Besides, the shape of the learning curves shows that the learning process is stable.

From Figure 10.6, it indicates that the validation accuracy nearly matches with the training accuracy. It means that the algorithm is confident in accurately classifying lesions to the corresponding categories during the training. Combining the results from Figures 10.4 and 10.5 demonstrates that the fine-tuned Xception model learns well and does overfit or underfit during the training process.

10.4.2 Classification Results

We use the prepared test set containing 2,534 images to evaluate the classification performance of our fine-tuned Xception model. It is noticed that this test set is completely separated from the training and validation sets so that the model is tested with unseen data.

The confusion matrix of our fine-tuned model is detailed in Table 10.4. As can be observed from this table, our model performs relatively well on the test set. The most successful predictions come from the NV class (the most popular one with the highest number of data samples), with 91% actual NV instances predicted as NV and the remaining 9% as BCC, BLK, or MEL. The following highest prediction results come with MEL class (the most dangerous type with the second-highest number of data samples), with 78% of samples being identified correctly. We perform data augmentation during the training process so that the number of samples is similar in each class. However, the test set is imbalanced because it is randomly extracted from the ISIC 2019 challenge dataset. Hence, in summary, although our model is trained on a balanced dataset, it can still predict an imbalanced dataset relatively well.

We summary the classification performance of our model for each class in Table 10.5. As depicted from this table, there is a clear difference between performance for each class since the dataset is vastly imbalanced. The best prediction result yields a 91% F1-score for the NV class, while the lowest is DF with only a

TABLE 10.4
Confusion Matrix on the Test Set

		Predicted (%)							
Actual		**AK**	**BCC**	**BKL**	**DF**	**MEL**	**NV**	**SCC**	**VASC**
	AK	51	14	0	0	21	0	14	0
	BCC	3	83	4	0	5	5	0	0
	BKL	2	4	53	2	17	20	2	0
	DF	0	0	0	71	0	29	0	0
	MEL	1	0	6	0	78	15	0	0
	NV	0	1	2	0	6	91	0	0
	SCC	11	6	0	0	11	11	61	0
	VASC	0	0	0	0	0	0	25	75

TABLE 10.5
Classification Result for Each Class

Class	Precision (%)	Recall (%)	F1-Score (%)
AK	47	57	52
BCC	80	78	79
BKL	72	54	62
DF	40	29	33
MEL	45	81	58
NV	93	89	91
SCC	62	44	52
VASC	75	75	75

33% F1-score. The performances are slightly better for AK and SCC classes than the DF class, yielding a 52% F1-score. These results show that the number of data samples in the test set has certain effects on the classification output.

From Table 10.5, it can also be seen that MEL class achieves much better recall than precision. This result can be explained as the similarity between MEL images and NV images in the test set. Figure 10.7 shows one example of similar images between MEL and NV classes. In general, all of the classes (except the DF class with the lowest number of samples in the test set) yield over 50% F1-score, which are acceptable results for multiple classifications of skin lesion images.

10.4.3 Comparative Study

This section compares our approach with the literature using the same ISIC 2019 dataset. Table 10.6 summarizes these result comparisons. From Table 10.6, our fine-tuned Xception model is a good classifier among other similar works on the same dataset. For example, our work obtains the best accuracy of 95.96% and the best BMA score of 64.3%. Regarding other performances such as precision, recall, and

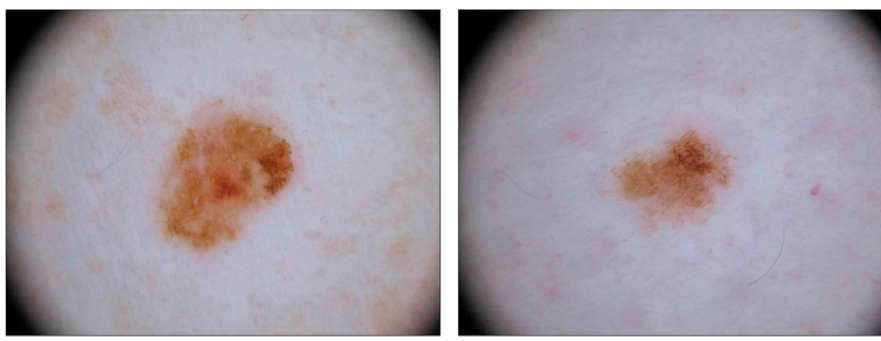

FIGURE 10.7 Example of the similarity between a MEL image (left) and an NV image (right).

TABLE 10.6
Performance Comparison With Other Works

Work	Method	Performance (%)				
		Accuracy	Precision	Recall	Specificity	BMA
Kassem et al., 2020 [22]	GoogleNet	94.92	80.36	79.8	97	–
Gessert et al., 2020 [21]ISIC 2019 Leaderboard rank 1	Multi-Res EfficientNets, SEN154 2	92.6	59.7	50.7	97.7	63.6
ISIC 2019 Leaderboard rank 2	EfficienetB3-B4-Seresnext101	91.7	50.7	60.7	95.2	60.7
ISIC 2019 Leaderboard rank 3	ResNet-152, DenseNet-201, SeResNext-101	92.4	58.4	54.0	96.3	59.3
ISIC 2019 Leaderboard rank 4	Ensemble 13 models	91.9	56.0	50.7	96.5	57.8
ISIC 2019 Leaderboard rank 5	Densenet-161	91.0	45.0	47.3	96.7	56.9
ISIC 2019 Leaderboard rank 6	Divide and conquer, Ensemble CNN networks	92.6	59.7	50.7	97.7	63.6
Ours	Xception	95.96	71.98	70.5	96.64	64.3

specificity, our model also gets quite good results compared to other works. It is also noticed that it is difficult to make a fair comparison since the test set of this work is randomly extracted from the ISIC 2019 dataset, thus different from other compared works in the literature.

10.5 CONCLUSION AND PERSPECTIVES

Skin cancer has raised attention in recent years, pressing needs to address this global general medical problem. This work presents a computer-aided diagnosis system capable of classifying eight different skin cancer types with reasonable accuracy by applying transfer learning and fine-tuning techniques to the pre-trained Xception model. We evaluate the model with dermoscopy images on a publicly available dataset named ISIC 2019.

Our method achieves 95.96% accuracy, 71.98% precision, 70.50% recall, 96.64 F1-score and 64.3% BMA score. These results show that our fine-tuned Xception model is a good classifier for the problem of multiple classifications of skin lesion images. This work can still go further if we have more data, especially for small classes like AK, SCC, VASC, and DF classes. Another direction is to continue improving the Xception model to deal with imbalanced dataset problems rather than just performing data augmentation to generate balanced classes.

REFERENCES

[1] Skin Cancer Foundation. Skin Cancer 101, Knowledge is Your Best Defense. Available online: https://www.skincancer.org/skin-cancer-information/. June 2021.

[2] Skin Cancer Foundation. Skin cancer facts & statistics. Available online: https://www.skincancer.org/skin-cancer-information/skin-cancer-facts/. June 2021.

[3] Vestergaard M. E., Macaskill P. H. P. M., Holt P. E., Menzies S. W. (2008). Dermoscopy compared with naked eye examination for the diagnosis of primary melanoma: a meta-analysis of studies performed in a clinical setting. *British Journal of Dermatology* 159(3):669–676. 10.1111/j.1365-2133.2008.08713.x.

[4] Lopez A. R., Giro-i-Nieto X., Burdick J., Marques O. (2017). Skin lesion classification from dermoscopic images using deep learning techniques. In: *IEEE 13th IASTED International Conference on Biomedical Engineering (BioMed'2017)*, pp 49–54. 10.2316/P.2017.852-053.

[5] Korotkov K., Garcia R. (2012). Computerized analysis of pigmented skin lesions: a review. *Artificial Intelligence in Medicine* 56(2):69–90. 10.1016/j.artmed.2012.08.002.

[6] Oliveira R. B., Papa J. P., Pereira A. S., Tavares J. M. R. (2018). Computational methods for pigmented skin lesion classification in images: review and future trends. *Neural Computing and Applications* 29(3):613–636. 10.1007/s00521-016-2482-6.

[7] Pathan S., Prabhu K. G., Siddalingaswamy P. C. (2018).Techniques and algorithms for computer aided diagnosis of pigmented skin lesions-a review. *Biomedical Signal Processing and Control* 39:237–262. 10.1016/j.bspc.2017.07.010.

[8] Murugan A., Nair S. A. H., Kumar K. S. (2019). Detection of skin cancer using SVM, random forest and kNN classifiers. *Journal of Medical Systems* 43(8):1–9

[9] Farooq M. A., Azhar M. A. M., Raza, R. H. (2016). Automatic lesion detection system (ALDS) for skin cancer classification using SVM and neural classifiers. *In 2016 IEEE 16th International Conference on Bioinformatics and Bioengineering (REF10_E)*, pp. 301–308. IEEE.

[10] Chaturvedi S. S., Tembhurne J. V., Diwan T. (2020). A multi-class skin Cancer classification using deep convolutional neural networks. *Multimedia Tools and Applications* 79(39), 28477–28498.

[11] Dorj U. O., Lee K. K., Choi J. Y. Lee M. (2018). The skin cancer classification using deep convolutional neural network. *Multimedia Tools and Applications* 77(8), 9909–9924.

[12] Hosny K. M., Kassem M. A., Foaud, M. M. (2018). Skin cancer classification using deep learning and transfer learning. In *2018 9th Cairo International Biomedical Engineering Conference (CIBEC)*, 90–93. IEEE.

[13] Tschandl P., Rosendahl C., Kittler H. (2018). The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific Data* 5, 180161. 10.1038/sdata.2018.161.

[14] Codella N. C. F., Gutman D., Celebi M. E., Helba B., Marchetti M. A., Dusza S. W., Kalloo A., Liopyris K., Mishra N., Kittler H., Halpern A. (2017). Skin lesion analysis toward melanoma detection: a challenge at the 2017 International Symposium on Biomedical Imaging (isbi), Hosted by the International Skin Imaging Collaboration (ISIC). arXiv:1710.05006.

[15] Combalia M., Codella N. C. F., Rotemberg V., Helba B., Vilaplana V., Reiter O., Halpern A. C., Puig S., Malvehy J. (2019). BCN20000: Dermoscopic lesions in the wild. arXiv:1908.02288.

[16] Gu J., Wang Z., Kuen J., Ma L., Shahroudy A., Shuai B., … Chen, T. (2018). Recent advances in convolutional neural networks. *Pattern Recognition* 77, 354–377.

[17] Kingma, D. P., Ba, J. (2014). Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980.
[18] Deng J., Dong W., Socher R., Li L. J., Li, K., Fei-Fei L. (2009). Imagenet: a large-scale hierarchical image database. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 248–255.
[19] Torrey L., Shavlik, J. (2010). Transfer learning. *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, 242–264. IGI global.
[20] Chollet F. (2017). Xception: deep learning with depthwise separable convolutions. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 1251–1258.
[21] Gessert N., Nielsen M., Shaikh M., Werner R., Schlaefer, A. (2020). Skin lesion classification using ensembles of multi-resolution Efficient Nets with meta data. *MethodsX 7*, 100864.
[22] Kassem M. A., Hosny K. M., Fouad, M. M. (2020). Skin lesions classification into eight classes for ISIC 2019 using deep convolutional neural network and transfer learning. *IEEE Access 8*, 114822–114832.