Journal of Korea Multimedia Society Vol. 23, No. 4, April 2020(pp. 525-538)
https://doi.org/10.9717/kmms.2020.23.4.525

Animal Face Classification using Dual Deep Convolutional Neural Network

Rafiul Hasan Khan[†], Kyung-Won Kang[††], Seon-Ja Lim[†††], Sung-Dae Youn[††††], Oh-Jun Kwon[†††††], Suk-Hwan Lee[††††††], Ki-Ryong Kwon[†††††††]

ABSTRACT

A practical animal face classification system that classifies animals in image and video data is considered as a pivotal topic in machine learning. In this research, we are proposing a novel method of fully connected dual Deep Convolutional Neural Network (DCNN), which extracts and analyzes image features on a large scale. With the inclusion of the state of the art Batch Normalization layer and Exponential Linear Unit (ELU) layer, our proposed DCNN has gained the capability of analyzing a large amount of dataset as well as extracting more features than before. For this research, we have built our dataset containing ten thousand animal faces of ten animal classes and a dual DCNN. The significance of our network is that it has four sets of convolutional functions that work laterally with each other. We used a relatively small amount of batch size and a large number of iteration to mitigate overfitting during the training session. We have also used image augmentation to vary the shapes of the training images for the better learning process. The results demonstrate that, with an accuracy rate of 92.0%, the proposed DCNN outruns its counterparts while causing less computing costs.

Key words: Animal Face Classification, Machine Learning, Batch Normalization, Exponential Linear Unit, Dual Deep Convolutional Neural Network.

1. INTRODUCTION

Animal recognition and classification have been considered as the forefront topic of computer vision. The problem with the automatic system is that, Animals do not appear individually at a scene rather it takes all the natural objects around it. Therefore, for the computer, it has to rectify the animal figure first and then classify it where humans can easily concentrate only on the animal. Besides, animals may appear under different illumination conditions, viewpoints, scales and shapes [1]. Methods for classifying human faces have been established with great accuracy but for the animal classifications, those methods have been proved far from accurate because of the diversity of animal classes with complex intra-class variability and inter-class similarity [1]. Researchers have at-

※ Corresponding Author : Ki-Ryong Kwon, Address: (48513) 45 Yongso-ro, Namgu, Busan, Pukyong National University, Korea, TEL : +82-51-629-6257, FAX : +82-51-629-6230, E-mail : kiryongkwon@gmail.com
Receipt date : Oct. 1, 2019, Revision date : Mar. 11, 2020
Approval date : Apr. 1, 2020
[†] Dept. of IT Convergence and Application Engineering, Pukyong National University (E-mail : raifulcse92@gmail.com)
[††] Dept. of Information and Communication Engineering, Tongmyong University (E-mail : kangkw@tu.ac.kr)
[†††] Dept. of Computer Engineering, Pukyong National University (E-mail : lsja76@gmail.com)
[††††] Dept. of Computer Engineering, Pukyong National University (E-mail : sdyoun@pknu.ac.kr)
[†††††] Dept. of Computer Software Engineering, Dongeui University (E-mail : ojkwon@deu.ac.kr)
[††††††] Dept. of Computer Engineering, Dong-A University (E-mail : skylee@dau.ac.kr)
[†††††††] Dept. of IT Convergence and Application Engineering, Pukyong National University
※ This research was supported by the Ministry of Trade, Industry and Energy for its financial support of the project titled “the establishment of advanced marine industry open laboratory and development of realistic convergence content”.

tempted several approaches to solve these problems with each one having its benefits and drawbacks. However, in recent years, Convolutional Neural Networks (CNN) based classification methods have gained a lot of attention. When the CNN layers go deep meaning if it has a large number of layers then it is called Deep Convolutional Neural Networks (DCNN). The examples of such networks are, Alexnet [2], GoogleNet [3], VGG19 [4], etc. However, these networks are fully concentrated on the classification task regardless of any specific kind. Some works like the one Tibor Trnovszky et al. introduced in [5] and Guobin Chen et al. introduced in [6] have concentrated on the animal classification. These networks followed the general forms of CNN and achieved a high accuracy rate.

So, inspired by all of these works, we decided to build our classification model which produces an astonishing accuracy rate comparing to its related works. Since there are high-end classification methods available in the market such as Alexnet [2], GoogleNet [3], VGG19 [4], etc, we decided to build a DCNN algorithm which will have low computing cost but high accuracy rate.

We began by collecting and modifying ten thousand animal faces of ten classes. Then we went on to build our dual DCNN, which is consisted of updated layers such as the Exponential Linear Unit (ELU) layer and Batch Normalization Layer. Usually, all the DCNNs at present use the Rectified Linear Unit (ReLU) but it constrains us from taking any non-zero values, which eliminates useful features to be analyzed by the DCNN. In addition, the use of the Batch Normalization layer enables the DCNN to calculate any large size dataset.

The notable feature of our model is that it is a dual DCNN network. Conventionally, CNNs are configured in a linear manner where all the layers of DCNN normally come one after another. The more we add layers to it, the more it gains accuracy. It sometimes leads to a complex and computationally high-cost algorithm. Therefore, we intended to build a network that would be efficient enough to operate as well as achieve the desired accuracy rate. Another purpose behind such action was to give our DCNN the higher ability that it needs to analyze large datasets as well as removes the notion of increasing the number of layers. We also trained our model with the Stochastic Gradient Descent with Momentum 'SGDM'. All of these modifications are advocated by the accuracy rate of our network, which outpaces the methods proposed by Tibor Trnovszky et al. [5] and Guobin Chen et al. [6]. This article discusses Tibor Trnovszky et al. [5] and Guobin Chen et al. [6] works in the related works section which is then followed by our proposed DCNN method, experimental results, and the conclusion.

2. RELATED WORKS

The image classification algorithm takes the image as input and returns output what the image contains [5]. CNN based Animal Classification System also shares the same principle. The CNN based Animal Classification System can be described in three steps i.e. the pre-processing step, the feature-learning step, and the classification step.

Firstly, in the pre-processing step, the input image goes through a rescaling and image augmentation process to maximize the effect of factors that influence the animal classification algorithm [5]. Secondly, in the feature learning step, the features of the input image are calculated using the convolution algorithm and at last in the classification step, it builds a predictive model using the features from the training data [7]. These predictive models use the learned features from the training data and compare them with the new data (test data/ validation data) to estimate their class labels [8]. The output classes are specific and the user can learn the exact name of the class based on the prediction

ratio.

The CNNs are a branch of neural networks that have been tested effectively in areas such as image recognition and classification. CNN has been successful in identifying animals, faces, objects and traffic signs apart from powering vision in robots and self-driving cars [5]. Here, we will discuss two classification methods based on DCNN, which solely concentrated on animal face classification. Later, we will discuss our proposed dual DCNN based animal face classification system.

2.1 DCNN of Tibor Trnovszky et al.

In this paper [5] Tibor Trnovszky, et al. discussed a fully connected DCNN dedicated to animal face classification. They built their network to classify five animal classes i.e. bear, hog, deer, fox, and wolf. The input image of their network contains 1024 pixels with a dimension of. For the explanation, they divided their DCNN into 8 blocks. All of the blocks are connected one after another.

In the first block, they used animal faces from the dataset as input data. To improve the computation time, animal faces were resized to 32×32×3 pixels while they were passing into the input layer. They used image augmentation to provide better experimental results. That means that the input data were scaled, rotated and shifted. In the second block, a 2D CNN layer was used that contains 16 feature maps with a 3×3 kernel dimension. This 2D CNN layer was then followed by a ReLU layer as an activation function. The output of the second

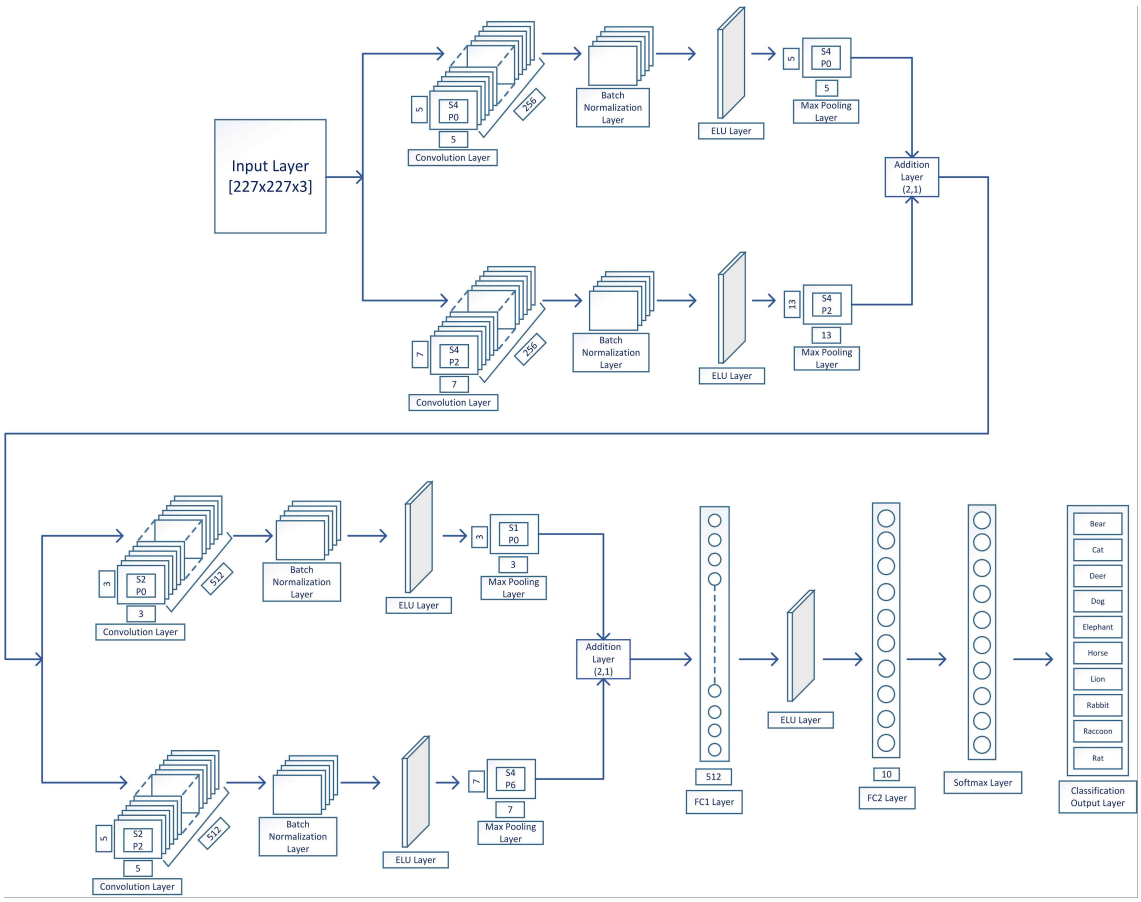

Fig. 1. Representation of the proposed method.

block goes into the third block, which contains a max-pooling layer. This layer contains a kernel dimension of 2×2 that subsequently followed by a dropout layer with a probability of 25%. They used a dropout layer to reduce the overfitting issue. Then in the fourth block, there is another set of 2D CNN layer and the ReLU layer. They used the same number of parameters as the second block except for this time the number of feature maps was doubled to 32. As like before, they used another Max Pooling layer and a Dropout layer with the same value as in block three in the fifth block. All the feature extraction layers finish here, which is then followed by a standard dense layer or popularly known as the fully connected layer in the sixth block. This dense layer has 256 neurons and a ReLU layer as an activation function. The output of the fifth layer turns into a vector and resides in these neurons. Then, the seventh block occupies a Dropout layer with a probability rate of 25%. At last, in the eighth block, the output dense layer or the fully connected layer was used for five animal classes with a softmax activation layer.

The notable feature of this network is that they used the ReLU layer as the activation layer and the dropout layer to reduce overfitting. There are some drawbacks of these layers such as ReLU does not allow non-zero values to be taken as features and some useful features may get overlooked due to the use of the Dropout layer. The structure of this network has been depicted in Fig. 2.

2.2 DCNN of Guobin Chen et al.

In this paper [6], Guobin Chen, et al. discussed a novel DCNN based species recognition algorithm for the wild animal classification on very challenging camera-trap imagery data. Their dataset contains a total of 20 animal classes with 14,346 images for training and 9,530 for testing purposes. The images of their dataset were captured with the motion-triggered camera. The moving foreground of the animals was selected as the region of interests and was fed to their DCNN based species recognition algorithm. They designed their DCNN with 3 convolutional layers and 3 max-pooling layers. All of the convolutional layers have a kernel with a size of 9×9 and pooling layers have a kernel with a size of 2×2.

Their DCNN begins with an input layer with a dimension of 128×128. Therefore, they scaled their dataset to 128×128 to match the input layer dimension. Right after the input layer, they added the first convolutional layer that applies a 2-D convolution to the 128×128 input layer. The output of the first convolution layer was a 120×120 matrix. Then they used a 2×2 max-pooling layer on the output of the first convolution layer. The first pair of convolution and max-pooling layers produce 32 feature maps of dimension, which were fed as the input for the second convolution layer. For each

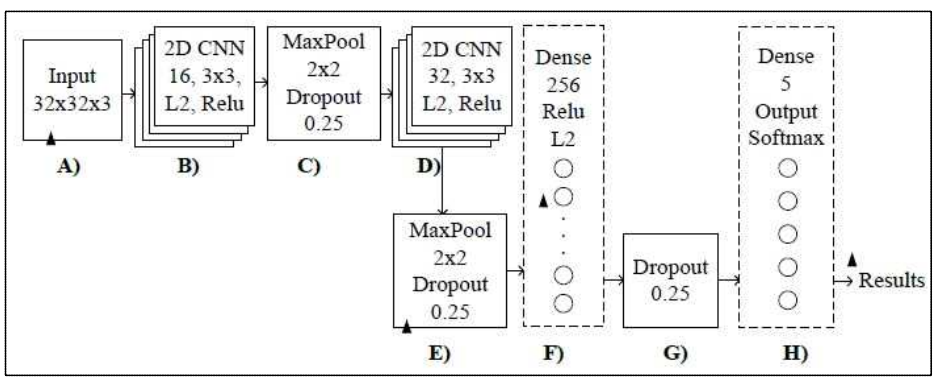

Fig. 2. The DCNN of Tibor Trnovszky et al. [5].

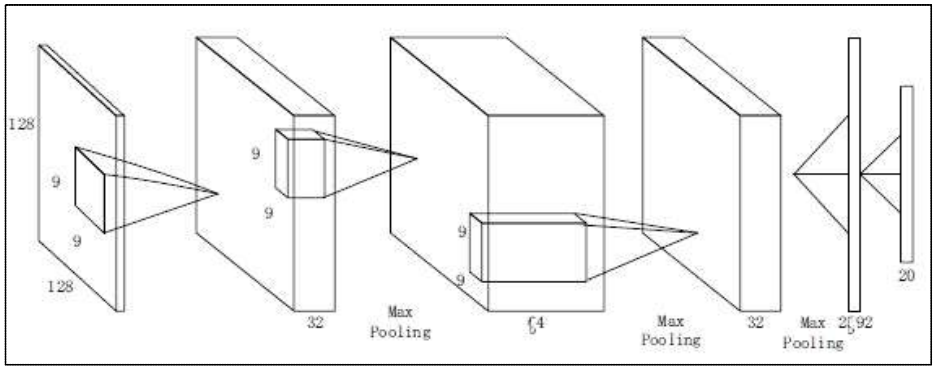

Fig. 3. The DCNN of Guobin Chen et al. [6].

kernel in the second convolution layer, they applied the convolution function to each input matrix and took the average to get an output matrix. The output of the second convolution layer was 64 feature maps with a dimension of 52×52. Then they used the second max-pooling layer that eventually produces 64 feature maps with a dimension of 26×26 for each of the matrices. As like before, the second max- pooling layer was followed by a third convolution layer, but this time the number of feature maps was reduced to half. They set 32 feature maps for the third convolution layer with a dimension of 9×9. Subsequently, the third pooling layer comes with a dimension of 2×2 and produces an output of 32 feature maps with a dimension of 9×9 matrices. After finishing the feature extraction process, the output goes to the first fully connected layer and turns into a vector. The dimension of the vector was 2592. This first fully connected layer was then followed by another fully connected layer and a softmax layer. The softmax layer has 20 neurons that determined the label of the input image. They also used a data augmentation [2] step during their training stage to bring variation in the learning process. The structure of this network has been depicted in Fig. 3.

One conspicuous point of their DCNN is that they did not use any activation layer. This means, all the features that were extracted by the convolution layers were analyzed. The drawbacks of such use are that it makes the whole system heavy as well as it is unnecessary to use all the features. Another point is that they did not use any dropout layer. There are both merits and demerits of using the dropout layer. It is advised to use a dropout layer while the DCNN works with a large dataset or it produces many features.

3. PROPOSED METHOD

The proposed dual DCNN can be divided into two parts- feature extraction layers and fully connected layers. In Fig. 4, Block 1 to Block 4 represents features extraction layers and the rest of it represents fully connected layers. There are four Convolution layers, four Batch Normalization layers, five Exponential Linear Unit (ELU) layers, four Max Pooling layers, two Fully Connected layers and one of each Input layer, Softmax layer, and Classification output layer.

Our proposed DCNN begins with an Input layer with a dimension of 227×227. Right after the Input layer, the DCNN follows four sets of feature extraction layers working dually with each other. For explanation, we have labeled those four sets of the feature extraction layers as Block 1, Block 2, Block 3, and Block 4. All of these blocks contain each of a Convolution layer, a Batch Normalization layer, an ELU layer, and a Max Pooling layer.

As shown in Fig. 4, Block 1, the Convolution

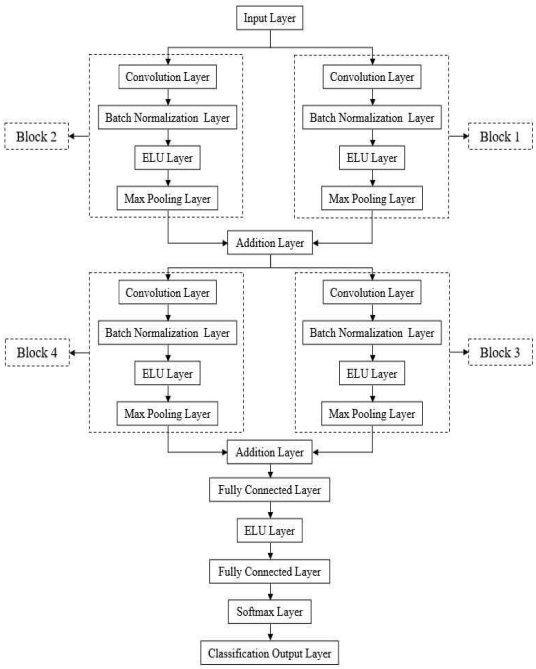

Fig. 4. Our proposed DCNN structure.

layer applies convolution function with a dimension of 5×5 and a stride of 4×4. This produces an output of 56×56×256. A Batch Normalization layer then normalizes this output. The normalized features are then forwarded to an ELU layer where the maximum number of useful features are extracted. Then, Max Pooling layer pools features from the output of the ELU layers with a dimension of 5×5 and a stride of 4×4. This produces an output of 13×13×256.

In Block 2, this same operation is followed except this time the Convolution layer applies convolution function with a dimension of 7×7 and a stride of 4×4. This produces an output of 57×57×256. Like Block 1, this output goes through a Batch Normalization layer and an ELU layer. After that, a Max Pooling layer applies pooling function with a dimension of 13×13 and a stride of 4×4 and produces an output of 13×13×256.

Eventually, an Addition layer elementwise adds the outputs of both of the Max Pooling layers from Block 1 and Block 2 and produces an output of 13×13×256. After that, it forwards the output to the next sets of the feature extracting layers that contains similar elements as the first set of layers except for this time the number of feature maps becomes double.

In Block 3, the Convolution layer produces 512 feature maps by applying a convolution function with a dimension of 3×3 and a stride of 2×2. This produces an output of 6×6×512, which is then normalized by a Batch Normalization layer. These normalized features then go through an ELU layer. As like before, this ELU layer makes sure that only the useful features are extracted and forwarded to the next layers. A Max-Pooling layer then applies pooling function with a stride of 1×1 and produces an output of 4×4×512.

On the other hand, Block 4 follows a similar operation by applying a convolution function with a dimension of 5×5 and a stride of 2×2. This operation produces an output of 7×7×512, which is then going through a Batch Normalization layer and an ELU layer. A Max-Pooling layer takes the output of the ELU layer and applies pooling function with a dimension of 7×7 and a stride of 4×4 and produces an output of 4×4×512.

Again, an Addition layer elementwise adds the outputs of both of the Max Pooling layers from Block 3 and Block 4 and produces an output of 4×4×512. At this point, the feature extraction operation is finished and ready to be forwarded for the first Fully Connected layer.

The first Fully Connected layer takes the output of the second Addition layer and turns it into a vector. This vector is then occupied by the 512 neurons of the first Fully Connected layer and then forwarded to the third and last ELU layer. After the application of ELU function, only the 10 best features are forwarded and occupied by the 10 neurons of the second Fully Connected layer. Then a Softmax layer applies a softmax function on the output of the second Fully Connected layer and finally forwards it to the Classification Output layer.

The Classification Output layer decides the most activated features and labels them with their corresponding class names. The whole operation has been depicted in Fig. 1.

3.1 Advantages of proposed dual DCNN

When we started building our DCNN, our goal was to create a network of low computing costs but high in the accuracy rate [9]. We have chosen dual DCNN because of its efficiency over linear DCNN. In addition, our proposed DCNN has been built with the state of the art layers i.e. Batch Normalization layer and Exponential Linear Unit (ELU) layer. Since we are dealing with a large dataset, our purpose was to generalize the dataset and take the strongest features by using those layers.

3.1.1 Dual DCNN vs Linear DCNN

The dual DCNN is more efficient and accurate than linear DCNN due to its structural configuration. The term 'dual' comes from its structure. It is apparent that in Fig. 4, Block 1 and Block 2 share the same input. They execute the feature extraction in a similar manner and in the end, they are combined by an Addition layer. This same operation is followed by Block 3 and Block 4. This operation resembles dualism thus we have labeled our proposed DCNN as dual DCNN. On the other hand, the works of Tibor Trnovszky, et al. [5] and Guobin Chen, et al. [6] are perfect examples of linear DCNN structure.

In linear DCNN, the layers are stacked one after another forming a linear shape for feature extraction. According to the hyper-parameter settings, these layers produce weights. Since it is one-way feature extraction, the DCNN only learns those features. For example, in the work of Guobin Chen, et al. [6], the first set of convolution layer and max-pooling layer produces 32 feature maps of dimension 60×60 by applying a convolutional kernel size of 9×9 and a pooling kernel size of 2×2. In this linear structure, the DCNN only have those weights to learn because there is no variety in the hyper-parameters settings.

On the other hand, in dual DCNN, the layers are designed with different hyper-parameters that produces a variety of weights. Because of it, the dual DCNN can learn a variety of features at the same time. For example, if we consider Fig. 4, both Block 1 and Block 2 may produce 256 feature maps of dimension 13×13, but the weights are different. The reason behind it is the different hyperparameter configurations. As it is mentioned before, Block 1 and Block 2 applies convolution function with a kernel size of 5×5 and 7×7 respectively. Therefore, the weights produced by them are also different. The Addition Layer elementwise adds those outputs and produces 256 feature maps of dimension 13×13, which is also completely different weight from the outputs of Block 1 and Block 2.

7. DATASET, EXPERIMENTS, AND RESULTS

Our whole expedition towards animal face classification started with the creation of the dataset. Then we went on to create our proposed dual DCNN. Below, we are showing the dataset creation process, which is then followed by the experimental results and findings.

4.1 Animal Face Dataset

We have collected images from different non-commercial sources, i.e. Google search and many other websites. Our dataset includes ten classes of animals i.e. bear, cat, deer, dog, elephant, horse, lion, rabbit, raccoon, and rat. Each of the animal classes contains 1000 images. When we collected those images, they came with many different backgrounds and with multiple other animals or images concentrating on full-body rather than the face. So, we had to modify them and took only the faces by cropping the images. There were also variations in animal faces. Such as, some photos

showed frontal views, some showed side views, etc. Since our goal was to work with the faces, so all the images were taken in the frontal position with the easiness for some side movements. Some animal images had a different dimension. So, all those images were resized to 227×227×3 since our network only takes images with the same dimension. The accuracy of the animal recognition system depends on the quality of the image dataset. The more different features it has, the more accuracy the animal classification system can achieve. Especially, the variance in color, edges, and corners will differentiate between classes. So, we maintained a refined quality while we were choosing the images for the dataset. Fig. 5. shows 40 example images from the created animal dataset.

4.2 Experimental Results

For the experiments, we have experimented and compared our work with the work of Tibor Trnovszky et al. [5] and Guobin Chen et al [6]. The compared works have been discussed in the above-mentioned related work section. In addition, to reach our proposed method, we went through several simulations. These simulations have helped us to decide which parameters or structure to choose. Among all of those simulations, we are discussing two structures below which we have labeled as "Linear DCNN + Batch Normalization + ELU" and "Proposed dual DCNN + Cross Channel Normalization + ReLU".

In "Linear DCNN + Batch Normalization + ELU", we have constructed a linear form of DCNN with the Batch Normalization layers and the ELU layers. In this structure, we have used two Convolution layers, two Batch Normalization layers, three ELU layers, two Max-Pooling layers, two Fully Connected layers and one of each Input layer, Softmax layer, Classification Output layer. Here, the first Convolution layer and the first Max-Pooling layer apply convolution function and pooling function with the same dimension of 5×5 and a stride of 4×4. Then, the second Convolution layer and the second Max-Pooling layer also applies similar convolution function and pooling function respectively with a dimension of 3×3 and a stride of 2×2. After that, a Fully Connected layer with 512 neurons forwards the output to the Classification Output layer through an ELU layer, a Fully Connected layer, and a Softmax layer. A graphical representation has been shown in Fig. 6.

In "Proposed dual DCNN + Cross Channel Normalization + ReLU", we have constructed our proposed dual DCNN with the conventional Cross-Channel Normalization layers and ReLU layers. The construction of this network follows our proposed dual DCNN except this one contains Cross-Channel Normalization layers and ReLU layers in-

Fig. 5. The example of the created animal database.

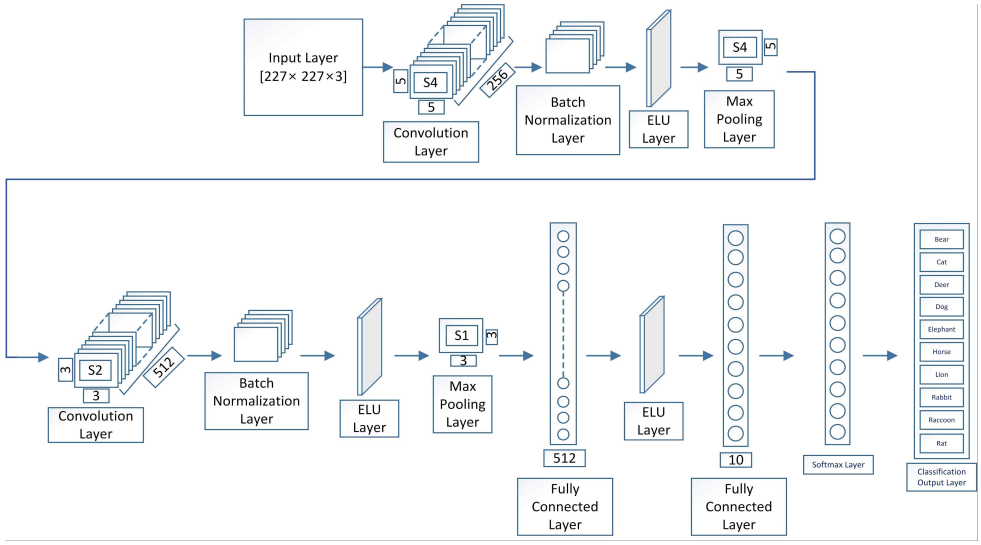

Fig. 6. "Linear DCNN + Batch Normalization + ELU" structure.

stead of Batch Normalization layers and ELU layers respectively. In this structure, we have used four Convolution layers, four Cross-Channel Normalization layers, five ReLU layers, four Max-Pooling layers, two Addition layers, two Fully Connected layers and one of each Input layer, Softmax layer, Classification Output layer. The window size of the Cross-Channel Normalization layer was set to 5. A graphical representation has been shown in Fig. 7.

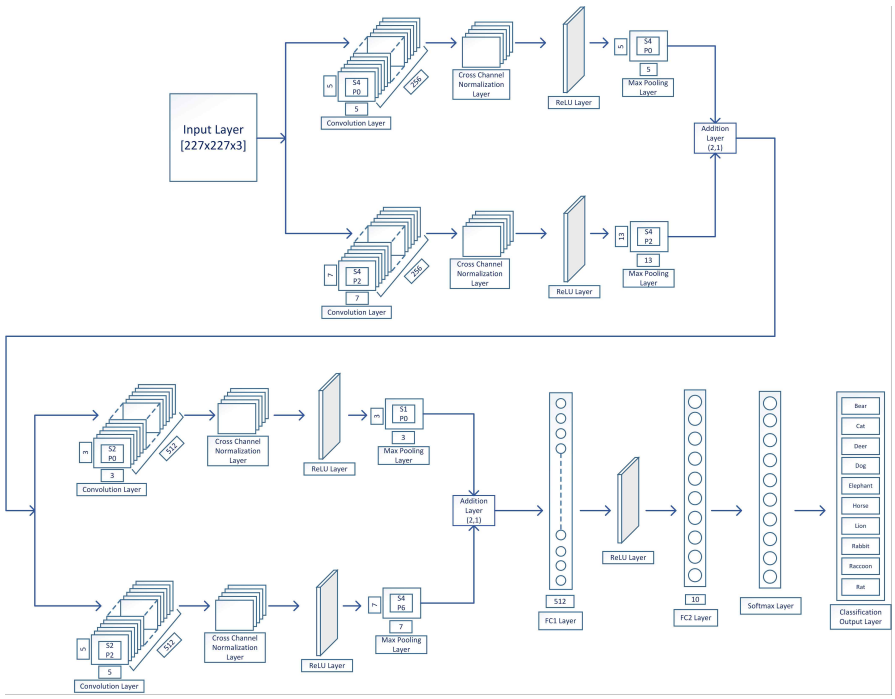

Fig. 7. "Proposed dual DCNN + Cross Channel Normalization + ReLU" structure.

For the simulation of all of these networks, we followed the same principle. We did simulations on all of these networks with our built-in dataset. The dataset was divided into eighty percent to twenty percent for training and validation/testing. The training dataset contained a total of 8000 training images and the validation/testing dataset contained a total of 2000 validation/testing images. After the division, we applied image augmentation on the training dataset. The image augmentation was used to augment images with a random combination of resizing, rotation, reflection, shear, and translation transformations which eventually boost the performance of the networks. Image augmentation brings diversity during the training process, and it gave a better learning environment for the DCNN. For all the simulations, we used the Stochastic Gradient Descent with Momentum (SGDM) and set an initial learning rate of 0.0001. We set max epoch to 300 and mini-batch size to 32 [10]. Shuffling also used for the dataset during training. Below, the simulation results of our proposed dual DCNN has been shown.

Table 1 displays the total validation accuracy along with the accuracy of all the individual classes. These validation accuracies are showing the results when we applied our trained DCNN on the twenty-percent validation/ testing dataset. Here, it has been shown that the overall accuracy rate was 92.0% while the Raccoon class had the highest accuracy rate with 97.5% and Cat class had the lowest accuracy rate with 85.0%. After we finished testing our proposed method, we went on to test the related works.

We followed the same network configuration of the related works and built them by ourselves. We kept an ideal training condition for all the networks as like our proposed DCNN. After finishing the

Table 1. Simulation results of our proposed dual DCNN method

Confusion Matrix												
		Target Class										
		Bear	Cat	Deer	Dog	Elephant	Horse	Lion	Rabbit	Raccoon	Rat	Total
Output Class	Bear	179 8.9%	0 0.0%	0 0.0%	2 0.1%	3 0.1%	2 0.1%	1 0.1%	0 0.0%	2 0.1%	0 0.0%	
	Cat	1 0.1%	170 8.5%	2 0.1%	5 0.3%	4 0.2%	0 0.0%	2 0.1%	1 0.1%	0 0.0%	2 0.1%	
	Deer	0 0.0%	5 0.3%	179 8.9%	2 0.1%	1 0.1%	0 0.0%	0 0.0%	2 0.1%	0 0.0%	5 0.3%	
	Dog	2 0.1%	13 0.7%	2 0.1%	182 9.1%	4 0.2%	5 0.3%	2 0.1%	4 0.2%	2 0.1%	2 0.1%	
	Elephant	1 0.1%	2 0.1%	5 0.3%	2 0.1%	185 9.3%	4 0.2%	0 0.0%	1 0.1%	0 0.0%	0 0.0%	
	Horse	8 0.4%	4 0.2%	6 0.3%	2 0.1%	1 0.1%	189 9.4%	1 0.1%	3 0.1%	1 0.1%	1 0.1%	
	Lion	2 0.1%	0 0.0%	1 0.1%	1 0.1%	1 0.1%	0 0.0%	194 9.7%	0 0.0%	0 0.0%	0 0.0%	
	Rabbit	1 0.1%	4 0.2%	2 0.1%	1 0.1%	1 0.1%	0 0.0%	0 0.0%	183 9.2%	0 0.0%	5 0.3%	
	Raccoon	0 0.0%	0 0.0%	2 0.1%	2 0.1%	0 0.0%	0 0.0%	0 0.0%	1 0.1%	195 9.8%	2 0.1%	
	Rat	6 0.3%	2 0.1%	1 0.1%	1 0.1%	0 0.0%	0 0.0%	0 0.0%	5 0.3%	0 0.0%	183 9.2%	
Total		89.5% 10.5%	85.0% 15.0%	89.5% 10.5%	91.0% 9.0%	92.5% 7.5%	94.5% 5.5%	97.0% 3.0%	91.5% 8.5%	97.5% 2.5%	91.5% 8.5%	92.0% 8.1%

building and training of all the related DCNNs, we tested them with the validation/testing dataset. The number of successfully recognized animals by all the related methods including our proposed method has been shown in Table.2. It is apparent that our proposed method recognized more animals than other related works. Where the most recognized class by the proposed DCNN of Guobin Chen et al [6] was Raccoon with 192 out of 200 and the least recognized class was Rat with 76 out of 200 images. On the other hand, the most recognized class by the proposed DCNN of Tibor Trnovszky et al [5] was also Raccoon with 175 out of 200 and the least recognized class was Dog with 112 out of 200 images. The most recognized class by the "Proposed dual DCNN + Cross Channel Normalization + ReLU" was also Raccoon with 190 out of 200 images and the least recognized class was also Deer with 165 out 200 images. At last, the most recognized class by the "Linear DCNN + Batch Normalization + ELU" was Lion with 192 out of 200 images and the least recognized class was Cat with 170 out 200 images.

Table 3 displays the overall accuracy of all the related methods along with our proposed method. As we can see, the Guobin Chen, et al [6] method achieved 68.8% and the Tibor Trnovszky, et al [5] method achieved 74.9%. Moreover, "Proposed dual DCNN + Cross Channel Normalization + ReLU" achieved 88.9%, "Linear DCNN + Batch Normalization + ELU" achieved 90.3%, and our proposed method achieved 92.0%.

With respect to the training time in Table 4, Guobin Chen, et al [6] method took the least time to train in 90 minutes and 16 seconds and our proposed method took the most time to train in 677 minutes and 43 seconds. Apparently, the training time increases as the hyper-parameter density increases. The training time may defer depending on the computer hardware configuration. Therefore, we set the accuracy rate as our evaluation criterion. Despite our proposed method took the

Table 2. Number of recognized animals by all the proposed and related DCNN methods

	Bear	Cat	Deer	Dog	Elephant	Horse	Lion	Rabbit	Raccoon	Rat
DCNN by Guobin Chen et al [6].	93	143	164	146	145	109	173	136	192	76
DCNN by Tibor Trnovszky et al [5].	131	145	174	112	164	164	171	148	175	114
Proposed dual DCNN + Cross Channel Normalization + ReLU	174	169	165	180	171	182	188	181	190	178
Linear DCNN + Batch Normalization + ELU	176	170	178	181	175	187	192	179	191	178
Our proposed dual DCNN Model	187	177	177	177	173	191	194	183	193	176

Table 3. Simulation results of all the DCNN networks

Methods	Accuracy%
DCNN of Guobin Chen et al [6].	68.8%
DCNN of Tibor Trnovszky et al [5].	74.9%
Proposed dual DCNN + Cross Channel Normalization + ReLU	88.9%
Linear DCNN + Batch Normalization + ELU	90.3%
Our proposed dual DCNN Model	92.0%

Table 4. Simulation time of all the DCNN networks

Methods	Training time
DCNN of Guobin Chen et al [6].	90 minutes and 16 seconds
DCNN of Tibor Trnovszky et al [5].	117 minutes and 49 seconds
Proposed dual DCNN + Cross Channel Normalization + ReLU	630 minutes and 05 seconds
Linear DCNN + Batch Normalization + ELU	292 minutes and 38 seconds
Our proposed dual DCNN Model	677 minutes and 43seconds

most time to train but according to the evaluation criterion, it is the most accurate method among all the related methods. Thus, we conclude that our proposed method is the most efficient method for animal face classification. This whole operation was done in a system that has Intel Core i7CPU 3.40 GHz, 8 GB RAM and a single NVIDIA GeForce GTX 1060 3GB GPU.

5. CONCLUSION

In recent years, the prevalence of deep learning methods especially, the Deep Convolutional Neural Network (DCNN) has made the performance of image classification state-of-the-art [11]. With the light of this fact, this paper has discussed the animal face classification using a dual DCNN. Compared to the conventional linear DCNN, we have proposed a new dual DCNN that included some of the new layers i.e. Batch Normalization layer, Exponential Linear Unit (ELU) layer. The effect of these included layers can be realized by comparing the output results. The superiority of Batch Normalization over cross-channel normalization can be seen from the results. In addition, the inclusion of the ELU layer gave our proposed dual DCNN an ample amount of features to learn. Which means, our proposed DCNN has learned better features than all the related methods. Furthermore, the creation of a dual network helped us to extract stronger features that would not be possible with the linear DCNN. Overall, it has been established that if we use the updated layers and a dual-structured DCNN, it performs better compared to the other methods.

In the future, it is possible to extend this work by modifying the DCNNs feature extraction ability by adding more layers or making the network larger. In addition, for any other classification purpose, the Batch Normalization can be used instead of cross-channel normalization and for better feature extraction capability, and the ELU layer can be used instead of ReLU or any other Activation layer.

REFERENCE

[1] S. Taheri, O. Toygar, "Animal classification using facial images with score-level fusion," *IET Computer Vision*, Vol. 12, Issue 5, pp. 679-685, Aug. 2018.

[2] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems,* pp. 1106-1114, 2012.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," *IEEE Conference on Computer Vision and Pattern Recognition,* Vol: 1, Pages: 1-9, 2015.

[4] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-scale Image Recognition," *arXiv preprint arXiv: 1409.1556,* 2014.

[5] T. Trnovszky, P. Kamencay, R. Orjesek, M. Benco, and P. Sykora, "Animal Recognition

System Based on Convolutional Neural Network," *Advances in Electrical and Electronic Engineering*, Vol. 15, No. 3, Sept. 2017.

[6] G. Chen, T.X. Han, Z. He, R. Kays, and T. Forrester, "Deep Convolutional Neural Network based Species Recognition for Wild Animal Monitoring," *IEEE International Conference on Image Processing*, pp. 858-862, Oct. 2014.

[7] Kernel (image processing). https://en.wikipedia.org/wiki/Kernel_(image_processing) (accessed Sep, 23, 2019).

[8] S. Mouloodi, H. Rahmanpanah, C. Burvill, and H. MS Davies, "Prediction of displacement in the equine third metacarpal bone using a neural network prediction algorithm," *Biocybernetics and Biomedical Engineering,* pp. 1-14, Sep. 2019.

[9] R. H. Khan, Y. Lee, S.H. Lee, O.J. Kwon, and K.R. Kwon, "Anthropomorphic Animal Face Masking using Deep Convolutional Neural Network based Animal Face Classification," *Journal of Korea Multimedia Society*, Vol. 22, No. 5, pp. 558 - 572, May 2019.

[10] Batch Normalization Layer. https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.batchnormalizationlayer.html (accessed Sep. 23, 2019).

[11] L.C. Ow Tiong, S. Tae Kim, and Y. Man Ro. "Multimodal Face Biometrics by Using Convolutional Neural Network," Journal of Korea Multimedia Society, Vol. 20, No.2, pp. 170-178, Feb. 2017.

Rafiul Hasan Khan

He received his B.S. degree in Electrical & Electronics Engineering at American International University Bangladesh, the People's Republic of Bangladesh in 2015. Since March 2018, he is a Master's student in the department of IT Convergence and Application Engineering at Pukyong National University. His research interests include Signal Processing and Machine Learning.

Kyung-Won Kang

He received a B.S., an M. S., and a Ph. D. degree in Electronics Engineering at Pukyong National University in 1996, 1998 and 2002 respectively. He worked at Homecat, a set-top box manufacture from 2006-2014. He is currently an assistant professor in the Department of information and Communications Engineering at Tongmyong University. His research interests are in the area of digital image processing and machine learning.

Seon-Ja Lim

She received a M. S. in Computer Education from Kyungsung University in 2001, and a Ph. D. candidate in Computer Engineering at Pukyong National University in 2007. She worked adjunct professor at Busan University of Foreign Language from 2006-2016. She is currently adjunct professor at Tongmyong University. Her research interests are in the area of big-data, digital image processing, artificial intelligence, and machine learning.

Sung-Dae Youn

He received a Ph.D. in Computer Science from Pusan National University in 1997. He is currently a professor in the Department of Computer Engineering at Pukyong National University from 1989. His research interests are in the area of parallel computing, multicasting, and big-data.

Oh-Jun Kwon

He received a B.S. in Electronic Engineering from Kyungpook National University, in 1986, an M.S. in Computer Science from Chungnam National University in 1992 and a Ph.D. in Computer Science from Pohang University of Science and Technology (POSTECH), Korea in 1998. From Jan. 1986 to Feb. 2000, he worked for the Electronic and Telecommunication Research Institute (ETRI). Since 2000, he has been a Professor of Computer and Software Engineering Department at Dongeui University. His research interests include a computer network, computer security, wireless internet, pattern recognition and machine learning/deep learning.

Suk-Hwan Lee

He received a B.S., an M.S., and a Ph.D. degree in Electrical Engineering from Kyungpook National University, Korea in 1999, 2001, and 2004 respectively. He is currently working as a Professor in the Department of Computer Engineering at Dong-A University. His research interests include multimedia security, digital image processing, and computer graphics.

Ki-Ryong Kwon

He received the B.S., M.S., and Ph.D. degrees in electronics engineering from Kyungpook National University in 1986, 1990, and 1994 respectively. He worked at Hyundai Motor Company from 1986-1988 and at the Pusan University of Foreign Language from 1996-2006. He is currently a professor in the Department of IT Convergence and Application Engineering at the Pukyong National University. He has researched the University of Minnesota in the USA in 2000-2002 with Post-Doc. and Colorado State University in 2011-2012 with visiting professors. He was the President of Korea Multimedia Society in 2015-2016. His research interests are in the area of digital image processing, multimedia security and watermarking, bioinformatics, weather radar information processing, and machine learning.