IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 10, No. 1, March 2021, pp. 84~92
ISSN: 2252-8938, DOI: 10.11591/ijai.v10.i1.pp84-92

Effect of data-augmentation on fine-tuned CNN model performance

Ramaprasad Poojary, Roma Raina, Amit Kumar Mondal
School of Enineering and IT, Manipal Academy of Higher Education, Dubai, United Arab Emirates

Article Info

Article history:

Received Jul 8, 2020
Revised Nov 1, 2020
Accepted Jan 28, 2021

Keywords:

Convolutional neural network
Fine-tuning
Data-augmentation
Deep learning
Keras

ABSTRACT

During the last few years, deep learning achieved remarkable results in the field of machine learning when used for computer vision tasks. Among many of its architectures, deep neural network-based architecture known as convolutional neural networks are recently used widely for image detection and classification. Although it is a great tool for computer vision tasks, it demands a large amount of training data to yield high performance. In this paper, the data augmentation method is proposed to overcome the challenges faced due to a lack of insufficient training data. To analyze the effect of data augmentation, the proposed method uses two convolutional neural network architectures. To minimize the training time without compromising accuracy, models are built by fine-tuning pre-trained networks VGG16 and ResNet50. To evaluate the performance of the models, loss functions and accuracies are used. Proposed models are constructed using Keras deep learning framework and models are trained on a custom dataset created from Kaggle CAT vs DOG database. Experimental results showed that both the models achieved better test accuracy when data augmentation is employed, and model constructed using ResNet50 outperformed VGG16 based model with a test accuracy of 90% with data augmentation & 82% without data augmentation.

Corresponding Author:

Roma Raina
Associate Professor at Manipal Academy of Higher Education
Dubai, United Arab Emirates
Email: roma.raina@manipaldubai.com

1. INTRODUCTION

Deep learning models & particularly deep neural networks have shown promising results when applied to computer vision problems such as image classification, recognition and object detection [1]. Most well-known deep neural network architecture used for computer vision problems is convolutional neural network (CNN), a class of artificial neural networks trained and tested successfully using ImageNet dataset [2]. CNNs utilize parameterized kernels for preserving image spatial characteristics [3-4]. Basic CNN architecture has a stack of convolutional layers, non-linear activation functions such as ReLU, pooling layers, Fully connected layers and softmax layers. For optimizing parameters of CNNs, they are trained on a larger dataset consisting of training samples or examples. CNNs have become more popular and have attracted more attention because of the increasing computing power, availability of low-cost hardware & open source algorithms [5-6].

According to researchers, deep learning techniques generate better results when they are trained using a larger dataset [7], whereas if the model is trained on a smaller dataset with few training samples, it may perform badly during validation and testing. This phenomenon is referred to as overfitting. This is because CNN memorizes irrelevant noise instead of detecting significant discriminative features [8]. This is a

very important issue in machine learning as the model is not generalizable on data which it has not seen during the training phase. Data augmentation is a very powerful technique for avoiding overfitting. Augmented data will present a more comprehensive set of possible points thus minimizing the distance between training, validation and test samples [1]. Overfitting can also occur when the model is very complex which may lead to high variance. In this case, regularization and drop out techniques are to be used to minimize overfitting problems [9]. Regularization technique will try to decay whereas drop out technique will try to minimize the complexity of the model by dropping neurons of hidden layers based on certain probability [10-11].

CNN models can be built using different approaches. Transfer learning and fine-tuning are the easiest and most efficient way of building CNN models whereby a model trained on a larger dataset is reused for a similar task having a smaller dataset. In the case of transfer learning, features learned from the pre-trained model are transferred to a new model without having to learn new features. In the case of Fine-tuning, along with transfer learning, transferred features are fine tuned to achieve better results. Transfer learning and fine-tuning methods are effective only when the new task is similar to the original task and has a smaller dataset [12-13]. CNN models can also be built to learn the features from scratch though it is not an easy task and demands high computing power for training the model. Deep learning models can be built, trained and validated using frameworks such as Keras, Tensorflow, Caffe, Pytorch and application such as Matlab deep network designer.

In this work, proposed CNN models are built by fine-tuning well-known pre-trained CNN models namely VGG16 and ResNet50. The training dataset is created using the Kaggle cat vs dog database and has two image classes. The data augmentation method is used to avoid overfitting by increasing the size of the training dataset. The effect of data augmentation is analyzed by comparing the training and validation accuracy curves of fine-tuned CNN models with and without data augmentation. Keras deep learning framework is used to build, augment data, and to train and validate the model. This section shows the organization of the paper. Section 1 presents a review of deep learning models, CNNs, challenges faced during training CNNs and solutions to overcome these challenges. Section 2 highlights related work on data augmentation, Section 3 indicates the research gaps and proposed solutions. Section 4 gives technical details of the proposed CNN models and data augmentation method. Section 5 highlights the results and the main contribution of the study. Section 6 concludes this work.

2. BACKGROUND AND RELATED WORK

The CNN model presented in this paper is derived from a pre-trained model trained on a larger dataset called ImageNet dataset. This section thus provides facts and figures of three well-known pre-trained CNN models. The section also reviews some of the work carried out on data augmentation techniques.

2.1. Pretrained CNN models

Alexnet: This was the first CNN model that has very few parameters compared to the other pre-trained CNN models and hence training requires less time. It consists of five convolutional and three fully connected layers and uses ReLU for the non-linear part. It has advantages such as fewer training parameters and strong robustness [14]. Alexnet gave Top-5 accuracy of 80.3% when trained on the Imagenet dataset.

VGG 16: This architecture has more number of layers when compared to Alexnet. It has a total of 41 layers comprising of convolutional and fully connected layers. It is also an improved version of Alexnet where large kernels are replaced by multiple 3x3 kernels. It is also originally trained on 1000 classes of images from the Imagenet dataset. An image size of 224 by 224 is used for training. It gave a top-5 accuracy of 92.3% when trained on ImageNet [15].

ResNet 50: Compared to Alexnet and VGG16, ResNet50 architecture has a different unit called Residual block. It is used to eliminate the vanishing gradient that occurs when the CNN model is trained using backpropagation. ResNet models are also used as a backbone for many computer vision tasks [16-17]. ResNet model has won the ImageNet challenge competition held during 2015. This network has a total of 50 layers.

2.2. Fine-tuned CNN models

M. Shamim Hossain *et al.* [14] in their work presented two different deep learning architectures. Firstly light model 6- layered CNN architecture is proposed; secondly, a fine-tuned visual geometry group-16 pre-trained deep learning model is presented. For evaluating the framework, the proposed paper uses two-color image datasets. The authors claimed that accuracies above 90% are achieved by these methods. Ling Zhu *et al.* [15] in his paper presented a deep learning model derived from Alexnet pre-trained model for the

classification of vegetables into five classes. The author uses Caffe deep learning framework to transfer the learned weights from Alexnet pre-trained model to customize the new model.

2.3. Data augmentation

Yu-Dong Zhang *et al*. [8] in his study, designed a 13-layer convolutional neural network (CNN) for fruit classification. This work uses three types of data augmentation namely image rotation, gamma correction, and noise rejection. The author has used stochastic gradient descent with momentum to train the CNN and having a batch size of 128. The author also claimed that the proposed method yields an accuracy of 94.94%. Agnieszka Mikołajczyk *et al*. [18] proposes a method of data augmentation based on image style transfer to pre-train the given neural network. The author claims that the new method improves the training process efficiency. Medical images are used for validating the proposed method.

3. RESEARCH GAPS AND PROPOSED SOLUTION

As seen from the literature review performed and presented in Section 2, this research aims at bringing the best features of the Fine-tuned model and data augmentation together to improve the performance of the proposed model. Specific research gaps addressed by this paper:

- M. Shamim Hossain *et al*. and Ling Zhu *et al*. built their CNN models by fine-tuning well-known pre-trained models such as Alexnet. As Alexnet already has shown a good result when pre-trained on the ImageNet dataset, the new model also will perform well. Fine-tuning also saves training time. The limitation of this paper is that data augmentation could have been used to improve the performance further.
- Yu-Dong Zhang *et al*. and Agnieszka Mikołajczyk *et al*. build models from scratch but used the data augmentation method to augment the examples in the training set. Data augmentation avoids overfitting and improves the performance of the model. Rather than learning the features from scratch, Transfer learning and fine-tuning will be better options when transferred features are learned using well-known CNN models and pre-trained on a larger dataset.

By considering the above research gaps in the literature, the overall aim of this paper is to build the CNN model by fine-tuning the pre-trained model and to use data augmentation to avoid overfitting. The research objectives of this paper are:

- Build and train the proposed CNN model by Fine-tuning VGG16 and ResNet50 pre-trained models on a custom dataset created using Kaggle CAT vs DOG database. A fine-tuned model can be trained using a system with limited computing resources.
- Obtaining enough training data is the biggest challenge when applying deep learning to real-world problems because it may be difficult or expensive. Hence, there is a need to increase the diversity of training set by incorporating the data augmentation method. It improves the performance of the model and avoids overfitting especially when the training set is smaller.
- Evaluate the performance of the fine-tuned model using Keras framework and determine the effect of data augmentation using training and validation accuracy and loss as the statistical parameters.

4. METHODOLOGY

The overall block diagram of this work is illustrated in Figure 1. The methodology is explained in the following sections:

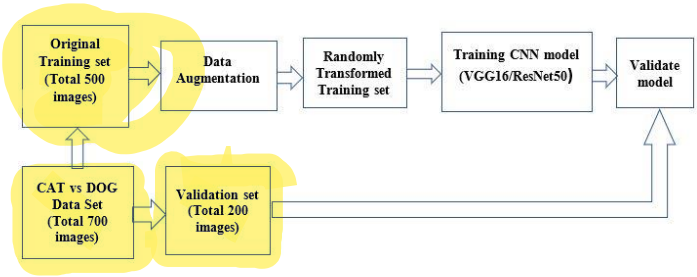


Figure 1. Overall block diagram of this work

4.1. Construction of proposed CNN models

In this work, proposed CNN model is built by fine-tuning previously trained networks VGG16 and ResNet50. These two models are originally trained on a larger dataset called ImageNet which has millions of images and can classify 1000 classes. The task of fine-tuning a network is to tweak the parameters of an already trained network so that it adapts to the new task at hand. During fine-tuning, only part of the network is trained and hence training takes less time. Since trainable parameters are less, the process does not demand very high computing resources. Fine-tuning multi-step process:

- Load the VGG16 or ResNet50 with weights pre-trained on the ImageNet dataset.
- Replace the original fully connected (FC) layers with a new one as per the given image class.
- Freeze all the CONV layers for transferring what has been learned.
- Train the network
- Unfreeze the last CONV block which extracts task-specific features.
- Train the network again to fine-tune parameters of the last CONV layer block.

Table 1 indicates the trainable and non-trainable parameters of fine-tuned models used for this study. As shown in the table fine-tuned VGG16 has 32,772,610 parameters to be fine-tuned during the training process. Trainable parameters include FC layers with 25,693,186 parameters and last conv block with 7,079,424 parameters.

Table 1. Number of trainable, non-trainable and total parameters for fine-tuned models

Parameters	Fine-tuned model using VGG16	Fine-tuned model using ResNet50
Total parameters	40,407,874	23,591,810
Trainable parameters	32,772,610	1,062,914
Non-trainable parameters	7,635,264	22,528,896

The table also shows the parameter breakup for the fine-tuned ResNet50 model. As shown in the last column of Table 1, there are 1,062,914 trainable parameters including 1,050,624 parameters of last conv block, 4098 parameters of FC layer, and 8192 parameters of the batch normalization (BN) layer. Batch normalization layers are used in ResNet50 to accelerate the convergence of CNN model. BN shifts the data points to the same scale and hence accelerate the training process. It also resolves the problems encountered during high learning rates [19]. In the case of BN, normalization is applied to each activation independently and hence the BN transform can be shown for a mini-batch as in (1).

$$y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)} \quad (1)$$

where $\hat{x}^{(k)}$ is the normalized value of $x^{(k)}$ and parameters $\gamma^{(k)}$ and $\beta^{(k)}$ are learned during the training process and are computed using (2-3) respectively. These parameters are used to scale and shift the normalized values [19-20] as shown in (1).

$$\gamma^{(k)} = \sqrt{Var[x^{(k)}]} \quad (2)$$

$$\beta^{(k)} = E[x^{(k)}] \quad (3)$$

4.2. Data Augmentation Factors

Often validation accuracy is lower than the training accuracy. Overfitting occurs when the gap between these two increases. This implies that instead of learning the significant image features, the CNN model attempts to learn the weights by heart. One way to prevent overfitting is to increase the diversity of training set [8]. It is achieved by data augmentation. Data augmentation extends the size of the training set by using factors such as rotation, shear, flipping, scaling, and translation. to increase the number of raw images. In this work, three factors (rotation, scaling and horizontal flip) are used to augment the training images.

- Horizontal flipping: This is more common than vertical flipping especially for datasets such as CATs vs DOGs. This augmentation is one of the easiest to implement and has proven useful on datasets such as CIFAR-10 and ImageNet. [1]. Vertical flip has no meaning when the dataset has images of CATs and DOGs.
- Rotation: This augmentation is done by rotating images in the clockwise or counterclockwise direction. In this study, random rotation up to 30º is made use of.

– Scaling or zooming: This augmentation zooms the image to make the image look smaller or bigger depends on the zoom value. Zoom value of less than 1.0 will reduce the size of the image and a value greater than 1.0 will increase the size. In this study, a zoom range between 0.75 and 1.25 is used.

4.3. Performance evaluation of proposed CNN models

4.3.1. Training and validation dataset

To evaluate the proposed model and to study the effect of data augmentation, a smaller custom dataset with two image classes (CATs and DOGs) is created from Kaggle CATs vs DOGs database by picking 700 images in total with 350 images of each class. As shown in Figure 1, the dataset thus created is split into training and validation set with 500 images and 200 images, respectively. Randomly transformed images are obtained by augmenting images in the training set by data augmentation and further used for training the proposed models. The performance of the model is validated to improve accuracy by using validation set. This study splits the dataset into 70% and 30% for training and validation, respectively.

4.3.2. Training of CNN models

Training is a process of finding optimal values for parameters of CONV, FC and BN layers of CNN such that actual outputs are close to desired outputs. The backpropagation algorithm is used to optimize the parameters based on the loss value generated during the forward pass. Several optimization algorithms are discussed in the literature and gradient descent is very powerful among them. To improve the performance of model, hyperparameters such as learning rate (LR), epochs, mini-batch size are carefully chosen [2].

4.3.3. Evaluation of CNN model

The model is used to predict the right classification label for images it has not seen. Actual labels generated by the classifier are compared with true classification labels of the image for improving the performance of the model or classifier [21]. The model is validated using images in the validation set which are not part of the training set. This study uses training and validation accuracy to evaluate the performance of the model.

4.3.4. Loss function

In deep learning, the objective function is referred to as a loss function and it indicates the error generated during the forward pass. The main purpose of training a model is to minimize the loss function by adjusting the parameters based on the loss function. Cross-entropy and mean square error (MSE) are the commonly used loss functions. Overall loss value for a given batch of training images is referred to as cost function. Equation (4) shows the categorical cross-entropy used as a loss function in this study. In this equation, P_y & P_f refer to ground-truth distribution and score distribution of 'x' respectively [22].

$$L(y, f(x)) = H(P_y, P_f) \triangleq -\sum_{i=1}^{n} P_y(x_i) \log P_f(x_i) \quad (4)$$

4.3.5. Mini-batch gradient descent

It is a version of stochastic gradient descent (SGD) with weight updates that take place for every minibatch rather than every sample. It converges faster than batch gradient descent as it performs forward pass for each mini-batch of N training samples and updates the weight during backpropagation. Since it reduces the variance of parameter-update, it leads to more stable convergence. Equations (5.a-b) describe the parameter update based on gradient descent algorithm with momentum [23].

$$m^{(i)} = gm^{(i-1)} - \eta \nabla L\left(\theta^{(i)}\right) \quad (5.a)$$

$$\theta^{i+1} = \theta^{i} + m^{(i)} \quad (5.b)$$

In these equations 'i' indicates the mini-batch number, 'η' indicates the learning rate, 'g' indicates the momentum value and $L\left(\theta^{(i)}\right)$ is the loss function. The proposed method uses an adaptive learning rate to speed up the training process and to ensure stable global minima.

5. RESULTS AND DISCUSSIONS

This section shows the results obtained during training, validating and testing the proposed models and indicates the important findings derived from these results. This work uses Keras [24] deep learning framework to construct, train, validate, and test the models. Keras is an API that works with Tensorflow as

the backend. All the experiments were carried out on a PC with Intel ® Core (TM) i5-6200U CPU machine with a frequency of 2.3GHz. As explained in Section 4, this study uses a mini-batch gradient descent optimization algorithm for adjusting the parameter values during backpropagation. In order the improve the performance of the model, various hyperparameters are chosen correctly as shown in Table 2. Learning rate (LR) is the important hyperparameters that influence the optimization's convergence and alters the cost function's curvature [25]. As shown in Table 2, adaptive learning rate with initial value and a decay rate is used in this study instead of a fixed learning rate. If fixed LR is selected, the algorithm may take too long to converge when LR is too small and may overshoot when LR is too large. Learning speed can also be increased by choosing the appropriate value for momentum during weight adjustment. A momentum value of 0.9 is an ideal value and used in this study. Table 2 also shows the mini-batch size and no. of epochs used for experimentation. As shown in the table, Initial LR for both models is selected as 0.1, and LR is updated using LR scheduler callback of Keras optimizer.

Table 2. Hyperparameters used for training fine-tuned VGG16 and ResNet50 models

Hyperparameters	Initial Learning Rate (LR)	LR Decay rate	Momentum	Min-batch size	No. of Epochs
Value	0.1	0.002	0.9	10	50

Figure 2(a-b) illustrate the training and validation processes for fine-tuned VGG16 models. As shown in Figure 2(a), without data augmentation, the model overfits on data. It is obvious from Figure 2(a), the margin between validation accuracy and training accuracy is high and increases. It is clear from the figure that, the model performs well on the training set but does not perform well on new data. It has been observed that the loss values also tend to increase during the validation process. Figure 2(b) shows the training and validation curves with data augmentation for VGG16 fine-tuned model. The graph shows that the model performs very well during training as well as validation. It is seen that validation accuracy stabilizes at 93% after 40 epochs.

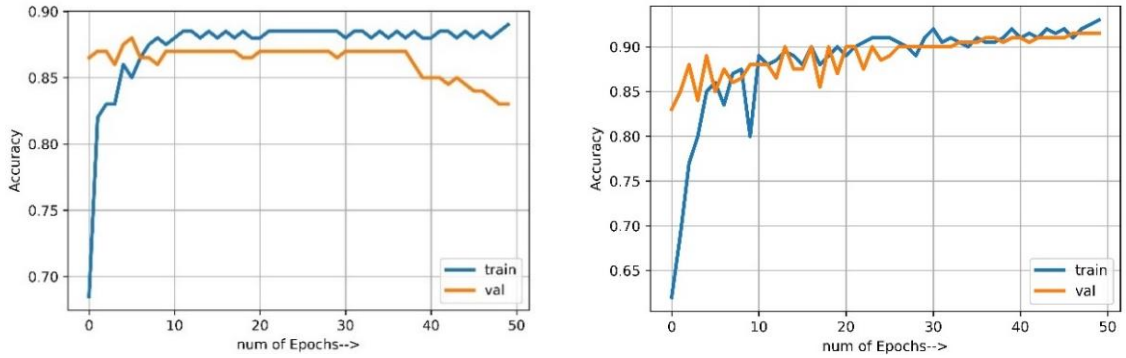
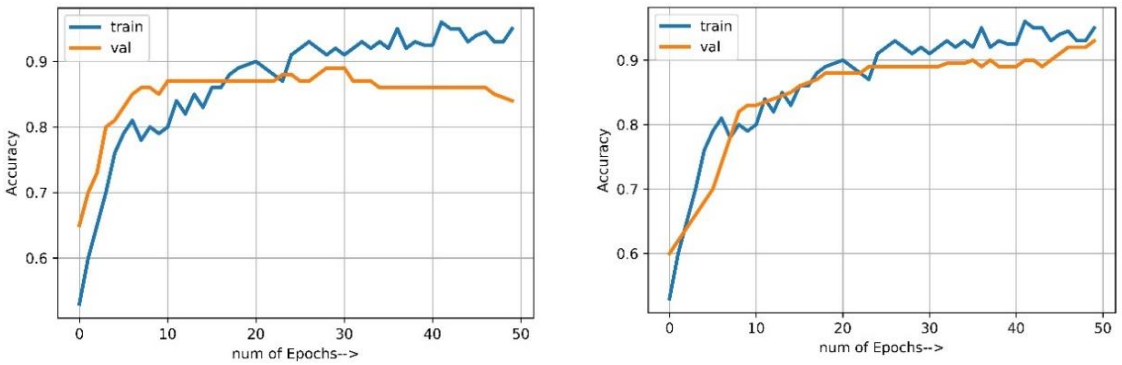


Figure 2. Training and validation accuracy curves of fine-tuned VGG16, (a) without and (b) with data augmentation

Figure 3(a-b) illustrate the training andd validation processes for fine-tuned ResNet50 models. Figure 3(a) shows the training and validation curves and it is obvious that model overfits as the model performs poorly on the validation set. Training accuracy is seen to be improving with the increase in epochs whereas validation accuracy is decreasing as number of epochs increases. Figure 3(b) shows that the model fits properly on new data and validation accuracy is seen to be following training accuracy, as the number of epochs increases.



(a) (b)

Figure 3. Training and validation accuracy curves of fine-tuned ResNet50, (a) without and (b) with data augmentation

5.1. Effect of data augmentation on accuracy values

Table 3 summarizes the impact of data augmentation on performance measures of both image classifiers. Final training accuracy and validation accuracy after 50 epochs is recorded as 93.5% and 91.5% for fine-tuned VGG16 image classifier with data augmentation. Final training and validation accuracy with data augmentation for fine-tuned ResNet50 model is slightly higher than that of VGG16 and recorded as 95% and 93% respectively. Accuracy values shown in the table are calculated as shown [26] in (6).

$$Accuracy = \frac{Number\ of\ correctly\ classified\ classes}{Total\ number\ of\ classes}\ x\ 100\% \quad (6)$$

As it is clear from the table that, with data augmentation, the model performs very well as the number of training samples will increase without changing the ratio of image classes.

Table 3. Effect of data augmentation on performance measures for fine-tuned models

Model	VGG16 fine-tuned model		ResNet50 fine-tuned model	
	With data augmentation	Without data augmentation	With data augmentation	Without data augmentation
Final training accuracy	93.5%	88%	95%	95%
Final validation accuracy	91.5%	83%	93%	84%
Test accuracy	88%	80%	90%	82%

As explained in Section 3, the categorical cross-entropy function is used to determine the loss value. Loss values are very less for models with data augmentation as the actual classes are very close to desired classes. The trained and validated model is tested using test data set which has 50 images of both classes. Images in the test data are not part of the training and validation set. Table 3 shows 88% test accuracy with data augmentation and 80% without data augmentation for the fine-tuned VGG16 model. Test accuracy for the fine-tuned ResNet50 model is slightly higher than that of VGG16 and recorded as 90% with data augmentation and 82% without data augmentation. Literatures also show that Top-1 accuracy of a pre-trained ResNet50 model, when trained on the ImageNet dataset, is higher than Top-1 accuracy of VGG16 model. Overall, model built from ResNet50 outperforms VGG16 based model.

5.2. Implementation of data augmentation

In this work, image data generator class of Keras deep learning framework is used for augmenting training samples. The number of training samples generated by data augmentation technique is very high when compared with training samples without the data augmentation when the number of epochs is high. In case of data augmentation, different series of training samples are generated during every epoch whereas, without the data augmentation, the same set of training samples are used during every epoch. Total number of training samples generated by the Image data generator for N epochs is calculated as given in (7).

$$M = Batch\_Size\ x\ Iterations\ x\ N \quad (7)$$

where 'M' indicates the total number of training samples generated by data augmentation in 'N' epochs. As per (7), total 2,500 random augmented images are generated by image data generator with 500 training samples, batch size of 10, and 50 epochs. Without data augmentation, 500 training samples are used during every epoch. Hence it is clear that data augmentation improves the diversity of the training set.

6. CONCLUSION

It is learned that, compared to building CNN models from the scratch, it is easier to build them from pre-trained models using fine-tuning and transfer learning if the given task is similar to the original task for which pre-trained models were trained on. In spite of having very good optimization techniques, the model may show poor accuracy during validation due to overfitting of data. To address the problem of overfitting, the proposed work used a data augmentation method which expands the size of the training dataset and increases the generalization of data. The proposed models are constructed by fine-tuning ResNet50 and VGG16 pre-trained models using Keras deep learning framework. Models are trained on a custom dataset created from Kaggle CAT vs DOG database. Mini-batch gradient descent optimizer is used for optimizing the model with an adaptive learning rate. For model evaluation, categorical cross-entropy is used as a loss function. Training and validation accuracies and training and validation losses are computed for analyzing the performance of the model. Test results showed that models with data augmentation outperform models without data augmentation. The main limitation of the work is since proposed models are built from pre-trained models with standard architectures, making changes to this architecture may severely affect the performance of the models.

REFERENCES

[1] Connor Shorten and Taghi Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," *Journal Big Data*, vol. 6, no. 1, 2019, DOI: 10.1186/s40537-019-0197-0.

[2] Yamashita, R., Nishio, M., Do, R.K.G. *et al*., "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, 9, pp. 611-629, 2018, doi: 10.1007/s13244-018-0639-9.

[3] Sambit Mahapatra,"Why Deep Learning over Traditional Machine Learning," Accessed on: Mar 21, 2018. [Online]. Available:https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063.

[4] Imane Hachchane, Abdelmajid Badri *et al*. "Large-scale image-to-video face retrieval with convolutional neural network features," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 9, no. 1, pp. 40-45, 2020, DOI: 10.11591/ijai.v9.i1.pp40-45.

[5] Arkadiusz Kwasigroch, Agnieszka Mikołajczyk, Michał Grochowski, "Deep neural networks approach to skin lesions classification - A comparative analysis," *22nd International Conference on Methods and Models in Automation and Robotics 2017, MMAR*, pp.1069-1074, 2017, doi: 10.1109/MMAR.2017.8046978.

[6] Abdelhafiz *et al*., "Dee convolutional neural networks for Mammography: advances, challenges and applications," *BMC Bioinformatics*, 20, 281, 2019, DOI: 10.1186/s12859-019-2823-4.

[7] Manisha Saini, Seba Susan,"Comparison of Deep Learning, Data Augmentation and Bag of-Visual-Words for Classification of Imbalanced Image Datasets," *Communications in Computer and Information Science book series CCIS*, vol. 1035, pp. 561-571, 2019, DOI: 10.1007/978-981-13-9181-1_49.

[8] Cagli E., Dumas C., Prouff E, "Convolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures", *International Conference on Cryptographic Hardware and Embedded Systems, CHES 2017*, vol. 10529, pp.45-68, 2017, doi:10.1007/978-3-319-66787-4_3.

[9] Truong Quang Vinh, Le Hoai Duy, Nguyen Thanh Nhan, "Vietnamese handwritten character recognition using convolutional neural network," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 9, no. 2, pp. 276-283, June 2020, DOI: 10.11591/ijai.v9.i2.pp276-283.

[10] Yim J., Ju J., Jung H., Kim J, "Image Classification Using Convolutional Neural Networks With Multi-stage Feature", *3rd International Conference on Robot Intelligence Technology and Applications*, pp. 587-594, 2015, DOI: 10.1007/978-3-319-16841-8_52.

[11] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Communications of the ACM, CACM 2017*, vol. 60, no.6, pp.84-90, June 2017, doi:10.1145/3065386.

[12] Hussain M., Bird J.J., Faria D.R, "A Study on CNN Transfer Learning for Image Classification," *18th Annual UK Workshop on Computational Intelligence 2018, UKCI 2018*, pp.191-202, 2018, doi:10.1007/978-3-319-97982-3_16

[13] Ramaprasad P, Akul Pai, "Comparative Study of Model Optimization Techniques in Fine-Tuned CNN Models", *2019 International Conference on Electrical and Computing Technologies and Applications, ICECTA*, pp.1-4, 2019, doi:10.1109/ICECTA48151.2019.8959681.

[14] Jing Sun, Xibiao Cai, Fuming Sun, Jianguo Zhang, "Scene image classification method based on Alex-Net model," *3rd International Conference on Informative and Cybernetics for Computational Social Systems, ICCSS*, pp. 363-367, 2016, doi: 10.1109/ICCSS.2016.7586482.

[15] Koustubh.: ResNet, AlexNet, VGGNet, Inception, "Understanding various architectures of Convolutional Networks," available: https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/.Last accessed 1st August 2019.
[16] Poonam Dhankhar, "ResNet-50 and VGG-16 for recognizing Facial Emotions," *International Journal of Innovations in Engineering and Technology, IJIET*, vol. 13, no. 4, pp. 126-130 July 2019, doi:10.21172/ijiet.134.18.
[17] Wonkyung Jung, Daejin Jung *et al*. "Restructuring Batch Normalization To Accelerate CNN Training," *2nd SysML Conference*, Palo Alto, CA, USA, 2019.
[18] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," *2018 International Interdisciplinary PhD Workshop, IIPhDW*, pp. 117-122, 2018, doi: 10.1109/IIPHDW.2018.8388338.
[19] Sergey Ioffe, Christian Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariance Shift," *32nd International Conference on Machine Learning*, vol. 37, 2015.
[20] Garbin, C., Zhu, X. Marques, O., "Dropout vs. batch normalization: an empirical study of their impact to deep learning," *Multimed Tools Appl*, vol. 79, pp. 12777-12815, 2020, doi:10.1007/s11042-019-08453-9.
[21] Mingyuan Xin and Yong Wang, "Research on image classification model based on deep convolutional neural network," *EURASIP Journal on Image and Video Processing*, 40, pp.1-11, 2019, doi:10.1186/s13640-019-0417-8.
[22] Sebastian Bruch, Xuanhui Wang, Michael Bendersky, Marc Najork, "An Analysis of the Softmax Cross Entropy Loss for Learning-to-Rank with Binary Relevance," *2019 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR*, pp. 75-78, September 2019, DOI: 10.1145/3341981.3344221.
[23] Timothy Anderson, "Regularization and Applications of a Network Structure Deep Image Prior'," *Semantic Scholar*, Corpus ID: 4498528, 2018, available:https://www.semanticscholar.org/paper/Regularization-and-Applications-of-a-Network-Deep-Anderson/9e902de448b43984703aac63cf279df3a9c444f7.
[24] F. Chollet, "Transfer learning and fine-tuning," Accessed on: August 13, 2020. [Online]. Available: https://keras.io/guides/transfer_learning/.
[25] Katanforoosh, Kunin *et al*., "Parameter optimization in neural networks", 2019. Accessed on: July 28, 2020. [Online]. Available: https://www.deeplearning.ai/ai-notes/optimization/.
[26] Khandakar M. Rashid, joseph Louis, "Times-series data augmentation and deep learning for construction equipment activity recognition," *Advanced Engineering Informaics*, vol. 42, pp. 1-12, October 2019, doi:10.1016/j.aei.2019.100944.

BIOGRAPHIES OF AUTHORS

Ramaprasad Poojary Ramaprasad Poojary is working as an Associate Professor at Manipal Academy of Higher Education Dubai Campus in Dubai, UAE. He has more than 20 years of Teaching and 6 years of research experience. He has pursued his B.E. in Electronics and Communication Engineering and M.Tech. in Digital Electronics & Communication and Ph.D. in Dental Image Registration. Presently he is working in Convolution Neural Networks.

Roma Raina Author currently works as an associate Professor in Manipal Acdemy of Hgher education, Dubai. She has more than 20 years of Teaching experience. Her area of interest lies in Controls & Instrumentation, Electrical Power systems. Her area of expertise lies in Radical Power Distribution, Electrical Machines. Author has published several research papers in her area of expertise.

Amit Kumar Mondal Author currently works as Assistant Professor in the Department of Mechatronics Engineering, Manipal Academy of Higher Education, Dubai, UAE. His area of research interest are Mobile Robotics, Autonomous System, Industrial Automation. Published more than 40 papers in national and international journals and conferences, filed 3 patents and successfully completed 3 externally funded projects from SERB, IUSSTF. He has received International travel support from SERB in the year 2015 and 2016.