EnsemV3X: A novel ensembled deep learning architecture for multi-label scene classification

Priyal Sobti [1] , **Anand Nayyar** [Corresp., 2] , **Niharika** [1] , **Preeti Nagrath** [1]

[1] Department of Computer Science and Engineering, Bharati Vidyapeeth's College of Engineering, New Delhi, India

[2] Graduate School / Faculty of Information Technology, Duy Tan University, Da Nang, Viet Nam

Corresponding Author: Anand Nayyar
Email address: anandnayyar@duytan.edu.vn

Convolutional neural network is widely used to perform the task of image classification, including pretraining, followed by fine-tuning whereby features are adapted to perform the target task, on ImageNet. ImageNet is a large database consisting of 15 million images belonging to 22,000 categories. Images collected from the Web are labeled using Amazon Mechanical Turk crowd-sourcing tool by human labelers. ImageNet is useful for transfer learning because of the sheer volume of its dataset and the number of object classes available. Transfer learning using pretrained models is useful because it helps to build computer vision models in an accurate and inexpensive manner. Models that have been pretrained on substantial datasets are used and repurposed for our requirements. Scene recognition is a widely used application of computer vision in many communities and industries, such as tourism. This study aims to show multilabel scene classification using five architectures, namely, VGG16, VGG19, ResNet50, InceptionV3, and Xception using ImageNet weights available in the Keras library. The performance of different architectures is comprehensively compared in the study. Finally, EnsemV3X is presented in this study. The proposed model with reduced number of parameters is superior to state-of-of-the-art models Inception and Xception because it demonstrates an accuracy of 91%.

Ensembled Model of InceptionV3 and Xception - EnsemV3X: Deep Learning Architecture for Multilabel Scene Classification

Priyal Sobti[1], **Anand Nayyar**[2], **Niharika**[1], **and Preeti Nagrath**[1]

[1]**Bharati Vidyapeeth's College Of Engineering New Delhi, Delhi, India**
[2]**Duy Tan University, Da Nang, Vietnam**

Corresponding author:
Anand Nayyar[2]

Email address: anandnayyar@duytan.edu.vn

ABSTRACT

Convolutional neural network is widely used to perform the task of image classification, including pretraining, followed by fine-tuning whereby features are adapted to perform the target task, on ImageNet. ImageNet is a large database consisting of 15 million images belonging to 22,000 categories. Images collected from the Web are labeled using Amazon Mechanical Turk crowd-sourcing tool by human labelers. ImageNet is useful for transfer learning because of the sheer volume of its dataset and the number of object classes available. Transfer learning using pretrained models is useful because it helps to build computer vision models in an accurate and inexpensive manner. Models that have been pretrained on substantial datasets are used and repurposed for our requirements. Scene recognition is a widely used application of computer vision in many communities and industries, such as tourism. This study aims to show multilabel scene classification using five architectures, namely, VGG16, VGG19, ResNet50, InceptionV3, and Xception using ImageNet weights available in the Keras library. The performance of different architectures is comprehensively compared in the study. Finally, EnsemV3X is presented in this study. The proposed model with reduced number of parameters is superior to state-of-of-the-art models Inception and Xception because it demonstrates an accuracy of 91%.

INTRODUCTION

Many problems, such as transferring our findings to a new dataset (Goyal and Benjamin, 2014), object detection (Ren et al., 2015), scene recognition (Herranz et al., 2016), have been extensively investigated due to network architectures measured against the ImageNet dataset (Deng et al., 2009) (Krizhevsky et al., 2012). Furthermore, any architecture that performs effectively on ImageNet is assumed effectual on other computer vision tasks as well (Voulodimos et al., 2018). If the second to last layers demonstrate satisfactory features, then networks, such as ImageNet, show acceptable performance with transfer learning (Kornblith et al., 2019). ImageNet is an extremely diverse dataset comprising of more than 15 million images that belong to 22,000 categories (Krizhevsky et al., 2012) which are structured according to WordNet hierarchy (Miller, 1995). WordNet contains more than 100,000 multiple words or phrases called "synsets" or "synonym sets," and ImageNet attempts to provide 1,000 images for each synset (Miller, 1998). ImageNet aims to provide researchers with sophisticated resources for computer vision. An annual competition named ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) (Russakovsky et al., 2015) is organized by the ImageNet team using a subset of the ImageNet database. The competition uses around 1.2 million images that belong to 1,000 classes (Krizhevsky et al., 2012).

Deep networks consisting of hidden layers that learn different features at every layer are required. The deep network with the maximum amount of data can optimize learning (Lohr, 2012). However, the issue is that it is difficult to get a huge labeled dataset for training (Masud et al., 2008). Even if you get the dataset it might take a substantial amount of time and cost to train a deep network. This is where the

concept of pre-training (Erhan et al., 2010) (Zoph et al., 2020) comes in. Notably, many models that have
been trained on powerful GPUs using millions of images for hundreds of hours are available. Existing
pretrained models have been processed on a sizeable dataset to solve problems similar to ours (Marcelino,
2018). We performed transfer learning after selecting the pretrained model to use. Transfer learning
transfers information from one domain to another and improves the learner (Pan et al., 2008) (Torrey
and Shavlik, 2010) (Weiss et al., 2016). We have two domains, namely, source D_S and target D_T with
their tasks T_S and T_T respectively. Transfer learning is defined as the process of improving the target
predictive function given domains and tasks by using the required information from D_S and T_S, where D_S
is not equal to D_T or T_S is not equal to T_T (Weiss et al., 2016). We used five architectures available in the
Keras (Ketkar, 2017) library, namely, VGG16, VGG19, ResNet50, InceptionV3, and Xception (Sarkar
et al., 2018). We also aim to compare the performance of each architecture on our chosen dataset. Many
applications of scene recognition are used in social media and the tourism industry to extract the location
information of images. Hence, objectives of this study are as follows:

- to investigate the different architectures that will be used with ImageNet weights to perform the task of multiclass classification
- to compare the performance of different architectures used to perform the task of transfer learning on the dataset consisting of multiple classes
- to solve the scene recognition task using ImageNet and then use it for multilabel scene classification
- to explore the performance of transfer learning in the problem using metrics and graphs obtained
- to design an ensemble using optimally performing models and improve image classification

The remainder of this paper is arranged as follows. The literature is reviewed in Section 2. The
methodology and dataset used in this study are introduced in Section 3. Learning curves are described in
Section 4. The results of this study are discussed in Section 5. Finally, the conclusion and future scope
are presented in Section 6.

LITERATURE REVIEW

LeNet-5 (LeCun et al., 1989) mainly describes convolutional neural networks (CNNs) as a stack of
convolutional layers, followed by fully connected layers that have been successfully used on various
datasets, such as MNIST, (Deng, 2012) and in the ImageNet classification challenge. ImageNet is a
large and diverse dataset widely used in machine vision tasks. ImageNet Large Scale Visual Recognition
Challenge (ILSVRC) is a challenge that is organized by the ImageNet team that uses a subset of the
ImageNet database i.e. 1.2 million images belonging to 1000 object classes. Krizhevsky et al. (2012)
trained a deep CNN ILSVRC-2010 (Berg et al., 2010) to categorize images belonging to the dataset and
achieved top-1 and -5 error rates of 37.5% and 17.0%. The methodology used a CNN comprising of
650,000 neurons and around 60 million parameters.The deep network used the AlexNet architecture,
which consists of five convolutional layers, followed by three fully connected layers with a final 1,000-way
softmax layer. The researchers also used two GPUs to train the network given that 1.2 million images are
excessively large for a single GPU.

Simonyan and Zisserman (2014) addressed depth, with is another very important facet of CNNs. The
researchers evaluated deep networks consisting of 19 layers known as VGG19, which is composed of a
stack of convolutional layers, followed by three fully connected layers and a softmax layer as the final
layer. VGG19 achieved 23.7% and 6.8% as the top-1 and -5 validation error rates, respectively, and 6.8%
as the top-5 test error rate.

Yosinski et al. (2014) in the year 2014 used a neural network trained on ImageNet and attempted to
show the transferability of features and subsequently demonstrated that feature transferability decreases
as the distance between initial and final tasks increases.

Le and Yang (2015) and Yao and Miller (2015) attempted to investigate the effect of various parameters
, such as convolutional layer depth, dropout layers, and receptive field size, on the accuracy in the Tiny
ImageNet Visual Recognition Challenge. The TinyImageNet dataset is a subset of the dataset used in
ILSVRC-2010 that consists of 200 object classes, which use 500 training images and 50 validation
and testing images. The study also increased the network depth and then applied techniques, such as
parametric rectified linear unit (PRELu) (Xu et al., 2015) and dropout (Srivastava et al., 2014), to the
model. The researchers used images that were not initially annotated whereby algorithms must suggest
labels to which images belong. The methodology achieved a final error rate of 0.444.

Szegedy et al. (2015) proposed a new architecture called Inception for ILSVRC 2014. The Inception architecture mainly consists of modules that are stacked one over the other and sometimes succeeded by max-pooling layers. The researcher also suggested the use of GoogLeNet, which is another aspect of the Inception architecture that consists of 22 layers and demonstrates a top-5 error rate of 6.67%.

He et al. (2015) proposed PReLU, which is a generalization of the standard rectified linear unit (ReLU). PReLU improved the model fitting without any extra cost and helped achiev a top-5 test error rate of 4.94%, thereby indicating a 26% improvement over GoogLeNet (winner of ILSVRC14).

Han et al. (2015) followed three steps and attempted to only maintain important connections and weights without degrading the accuracy. The researchers used the ImageNet dataset with AlexNet and VGG-16 Caffe (Jia et al., 2014) models to show how reduced the number of parameters 9 and 13 times without deteriorating the accuracy.

Sun (2016) implemented a different version of ResNet with 34 layers. Huang et al. (2016) developed an improved model as the baseline after applying data augmentation and stochastic depth and then compared the results with those of other residual networks. Bloice et al. (2017) showed that heavy image augmentation significantly improves the accuracy with an error rate of 34.68%.

Simon et al. (2016) put forward a set of pretrained models, including ResNet-10, ResNet-50, and batch normalized versions of AlexNet and VGG19. Such models can be trained within minutes using powerful GPUs (Akiba et al., 2017). All their models performed better than previous models with Top-1 and -5 error rates, which are equivalent to 26.9% and 8.8% for VGG19, respectively.

Huh et al. (2016) investigated the importance of ImageNet in learning acceptable features and explored the behavior of ImageNet by fine-tuning three tasks, namely, object detection using PASCAL VOC (Everingham et al., 2010) 2007 dataset, action classification on PASCAL-VOC 2012 dataset, and scene classification on the SUN dataset (Xiao et al., 2010).

Bastidas (2017) benchmarked on the Tiny ImageNet challenge by adapting and fine tuning two such models , namely, InceptionV3 and VGGNet (Wang et al., 2015).

Transfer learning extracts features from trained ImageNet networks. Extracted features are further used to train SVMs (Mayoraz and Alpaydin, 1999) and logistic regression classifiers (Donahue et al., 2014) (Sharif Razavian et al., 2014)(Smola and Schölkopf, 1998) and showed great performance on tasks that were different from ImageNet classification. Kornblith et al. (2019) showed that ImageNet architectures generalize appropriately across datasets by using 16 networks with a top-1 accuracy range of 71.6% to 80.8% in ILSVRC 2012.

Our study aims to address the following research gaps and issues:

- Transfer learning improved the scene classification accuracy (Akilan et al., 2017) but some problems were still identified during analysis.

  - A considerable amount of training data are required when the deep CNN model containing a large number of parameters is trained using transfer learning.

  - Satisfactory results are difficult to obtain due to the high complexity of rich and dense indoor images.

- Images with rich semantic content pose a problem (Liu and Tian, 2019) when attempting to solve the scene classification problem.

ENSEMV3X: - ENSEMBLE MODEL OF INCEPTIONV3 AND XCEPTION

Face recognition (Jain et al., 2020), object detection(Goyal and Benjamin, 2014), and scene classification are application areas of CNN (Khan et al., 2018) (Koushik, 2016) (O'Shea and Nash, 2015). CNN comprises three layers, namely, convolutional layer, pooling layer and a fully connected layer. CNNs use the concept of learnable kernels in the convolutional layer to form the base for CNN. The convolutional layer produces a 2D activation map (Krizhevsky et al., 2012) and as per the stride value, we glide through the input to produce a scalar product for each value in the kernel. Forward pass of convolutional layer (Liu et al., 2015) can be described by equation 1.

$$x_{i,j}^l = \sum_m \sum_n w_{m,n}^l o_{i+m,j+n}^{l-1} + b_{i,j}^l, \tag{1}$$

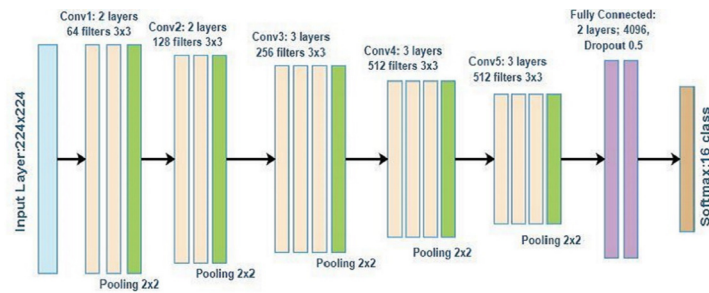



Figure 1. VGG16 Architecture

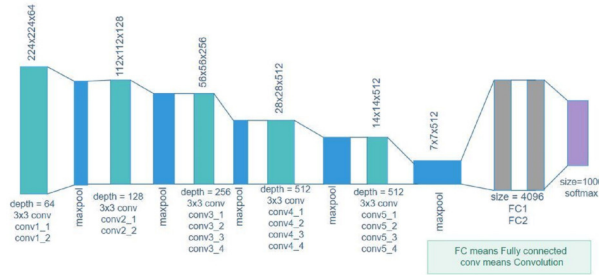



Figure 2. VGG19 Architecture

where x is the neuron output for row i and column j, w represents the weights, and b represents the biases (Gordon and Desjardins, 1995) for the lth convolutional layer. The number of parameters is reduced with the help of pooling layers. The subsequent application of an activation function (Sibi et al., 2013) accelerates up the learning process by adding non-linearity. Maxout, tanh, ReLU, and variants of ReLU, such as ELU, leaky ReLU, and PReLU (Xu et al., 2015) , help add nonlinearity. Fully connected layers (Basha et al., 2020) are then used to help determine optimal weights using backpropagation. These layers take the input from previous layers and analyze the output of all previous layers (Khan et al., 2019). Fully connected layers perform linear and nonlinear transformation on the incoming input. Equation 2 denotes the equation for linear transformation.

$$z = W^\top \cdot X + B, \tag{2}$$

139   where X represents the input; W is the weight; and B is the bias, which is a constant.
140   We investigate and implement different ImageNet architechtures, such as VGG16, which was used to
141   win the ILSVRC 2014, in this study. It uses a large number of hyperparameters of approximately 138
142   million and a 3×3 filter for a stride of one for convolutional layers and a 2×2 filter for a stride of two for
143   the maxpool layer. This structure is followed throughout the architecture. The number 16 in this study
144   denotes the number of layers with weights. Figure 1 shows the architecture of VGG16 consisting of two
145   convolutional blocks with two layers each, three convolutional blocks with three layers, and two fully
146   connected layers. The number of filters increases with the increase of convolutional blocks. Hence, the
147   first block has 64 filters with a size of 3×3, the second block has 128 filters, the third block has 256 filters,
148   and the last two blocks have 512 filters with a size of 3×3. The VGG16 architecture used in this study
149   classifies the output into 16 classes.
150   Simonyan and Zisserman (2014) further suggested that VGG19 is a deep network with a large number
151   of layers. The image input with a size of 224×224 for this model was trained on more than a million
152   images belonging to 1,000 object categories. VGG19 successfully learned many features from a wide
153   range of images. Figure 2 depicts the VGG19 architecture consisting of five blocks of convolutional
154   layers, followed by two fully connected layers. The number of filters for the first block are 64 followed by
155   128 and 256 filters, and the last two blocks have 512 filters with a size of 3×3. Figure 2 is the general
156   representation of the VGG19 architecture.
157   ResNet50 is generally used as the model for transfer learning and a miniaturized version of ResNet152.
158   ResNet (He et al., 2015) was the winner of the ImageNet challenge in 2015 and has since been used for

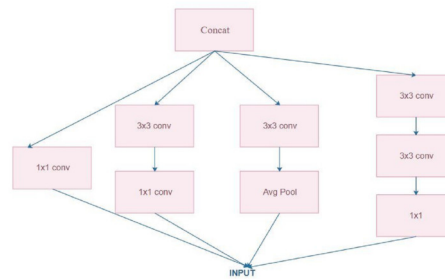



Figure 3. Inception Architecture

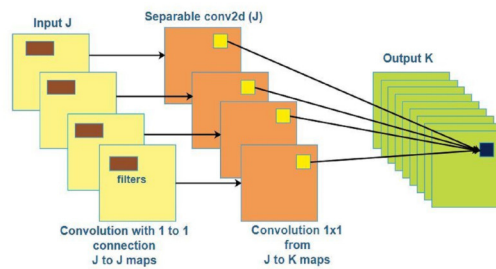



Figure 4. Xception Architecture

a number of tasks related to computer vision. This model helps to overcome the problem of vanishing gradient (Hochreiter, 1998) and allows us to train neural networks that are more than 150 layers deep. Skip connection (Furusho and Ikeda, 2019) is the main innovation in ResNet that allows the skipping of some irrelevant layers for training.

Szegedy et al. (2015) introduced the Inception model for ILSVRC14. Inception V3 is a CNN with 48 layers given by Google and trained on nearly 1 million images that classifies images to 1,000 classes, such as mouse, pencil, keyboard and a number of animals. Figure 3 depicts the architecture of the Inception model. Convolutional layers (1×1, 3×3, and 5×5) are used and their output is concatenated to form the input in the next stage. Convolutional layers with a size of 1×1 are used in this study to reduce the dimensionality (Van Der Maaten et al., 2009). This inception module is thus used in large architectures. Such a concept is mainly used to highlight the important learnings from previous layers for all subsequent layers (Stone and Veloso, 2000).

Finally, Xception is another CNN that is 71 layers deep and requires an image with an input size of 299x299. Francois Chollet (Chollet, 2017) proposed an original deep CNN inspired by Inception and named it Xception. Xception performs better than InceptionV3 on the ImageNet dataset and performs significantly better on large image classification datasets with nearly 350 million images and 17,000 classes. Figure 4 depicts the architecture of the Xception model.

Dataset

We used a dataset published by Intel to host an image classification challenge. The dataset extracted from Kaggle and is called **"Intel Image Classification,"**(Intel, 2018) consisting of 25,000 images under six labels, namely, building, glacier, mountain, forest, sea, and street. We utilized the entire dataset for training the model and subsequently tested it. The dataset has around 14,000 images for training, 3,000 for testing, and 7,000 for prediction.

Methodology

We used the concept of transfer learning in this study. Transfer learning (Weiss et al., 2016) helps build deep neural networks in a fast and accurate manner due to the usage of pretrained networks that can later be used for specific tasks.

CNNs consist of a convolutional base and a classifier. The convolutional base helps extract features from images using pooling and convolutional layers while the classifier helps label the image on the

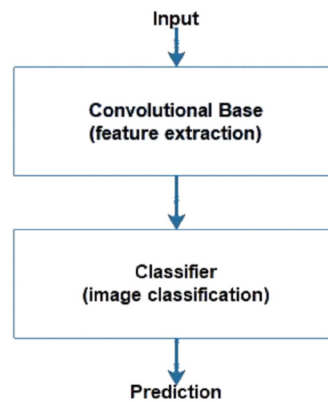



Figure 5. Components of CNN

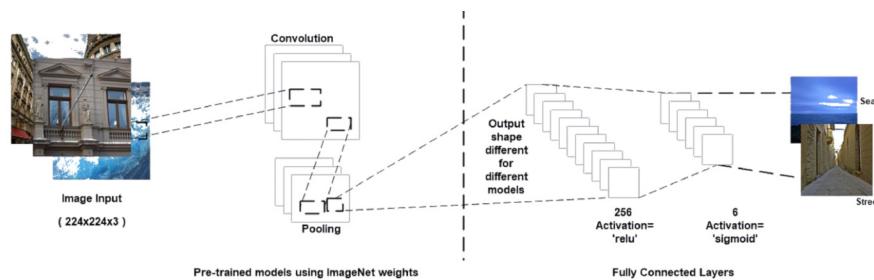



Figure 6. Proposed Methodology

Figure 7. EnsemV3X

(a) Training vs Validation Accuracy   (b) Training vs Validation Loss

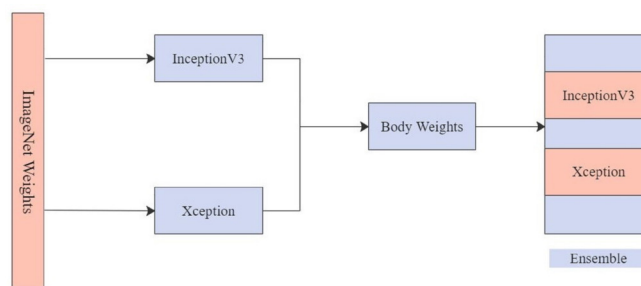
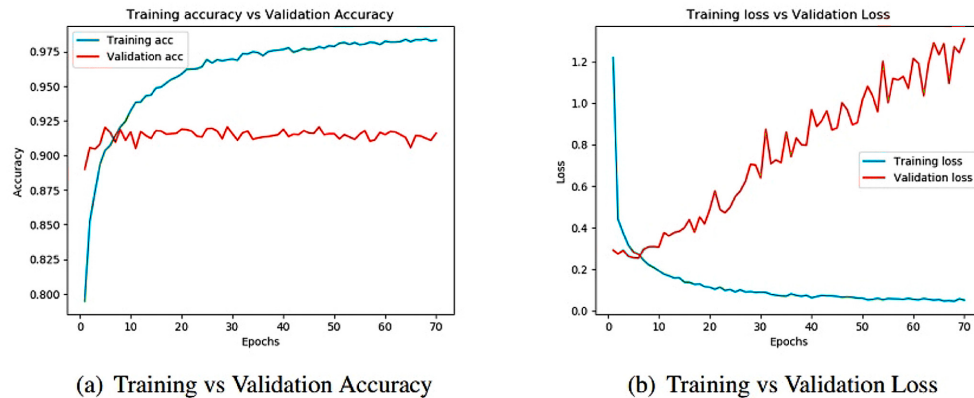



Figure 8. Training and Validation Metrics for InceptionV3

(a) Training vs Validation Accuracy   (b) Training vs Validation Loss

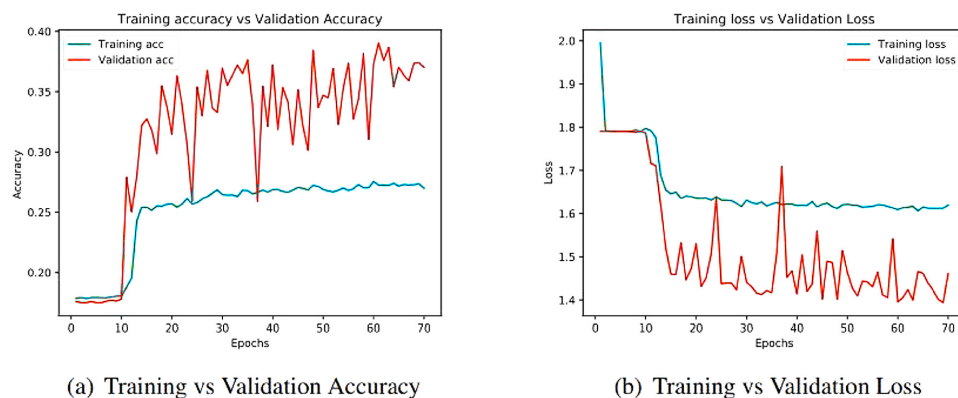



Figure 9. Training and Validation Metrics for ResNet50

basis of recognized features. Figure 5 shows the two components of CNNs. Keras comes bundled with a number of models , such as VGG, ResNet, Inception, Xception (Sarkar et al., 2018), and others. These models generally have two parts, namely, model architecture and weights (Gulli and Pal, 2017). Model weights are excluded from Keras but can be included using the weight parameter in the model definition. We used the weight parameter in the model definition to import ImageNet weights for each model during implementation. Figure 6 shows the flowchart for the implementation. We performed the task of multiclass classification (Aly, 2005) to analyze deep features of various scenes, such as forest, sea, or street, using models bundled with Keras. We started with a pretrained model by importing it from the Keras library and then setting the weight parameter in the model to " imagenet." Images were converted to appropriate dimensions according to the model and then features were extracted from images to form a part of the convolutional base. We used fully connected layers as the classifier to classify images to their respective classes depending on the probability of each class.

The top two models, Xception and InceptionV3, were chosen for EnsemV3X after obtaining the results. The two models are used to extract image features from the dataset, and their individual weight files are saved after training images. These weight files are then loaded into the respective classifiers of the two architectures and, the models are used to create an ensemble. Figure 7 shows the ensemble model created from Xception and InceptionV3. Weight files obtained by running the models for InceptionV3 and Xception were then used to train the ensemble and obtain the results for the test data using the ensemble model.

(a) Training vs Validation Accuracy   (b) Training vs Validation Loss

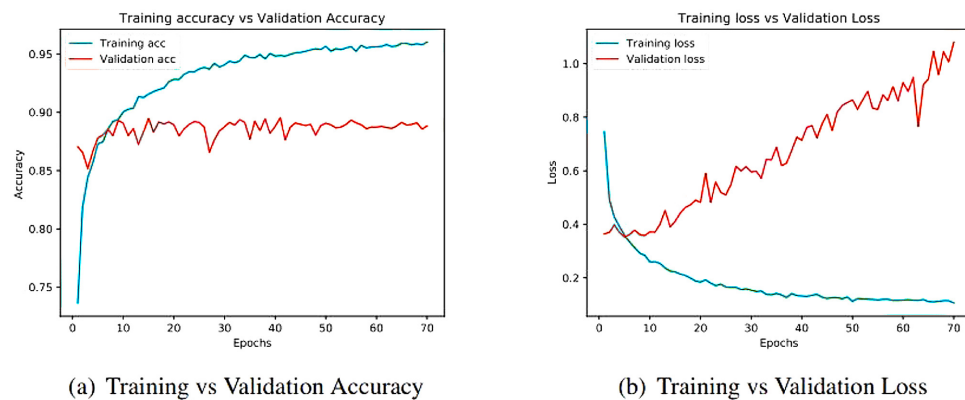



Figure 10. Training and Validation Metrics for VGG16

(a) Training vs Validation Accuracy   (b) Training vs Validation Loss

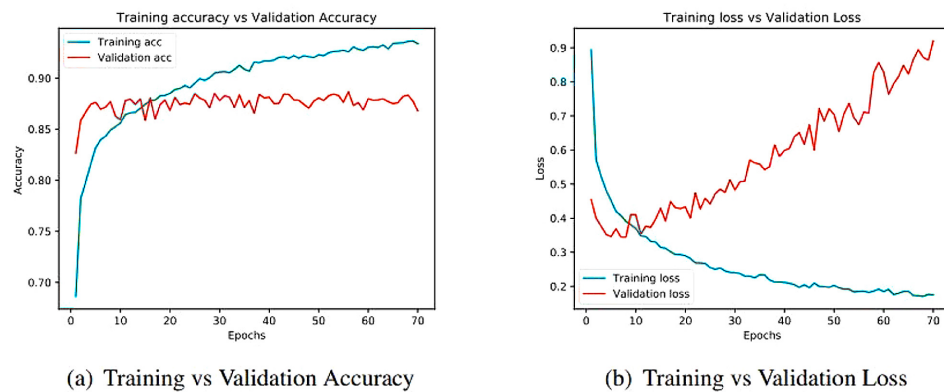



Figure 11. Training and Validation Metrics for VGG19

(a) Training vs Validation Accuracy   (b) Training vs Validation Loss

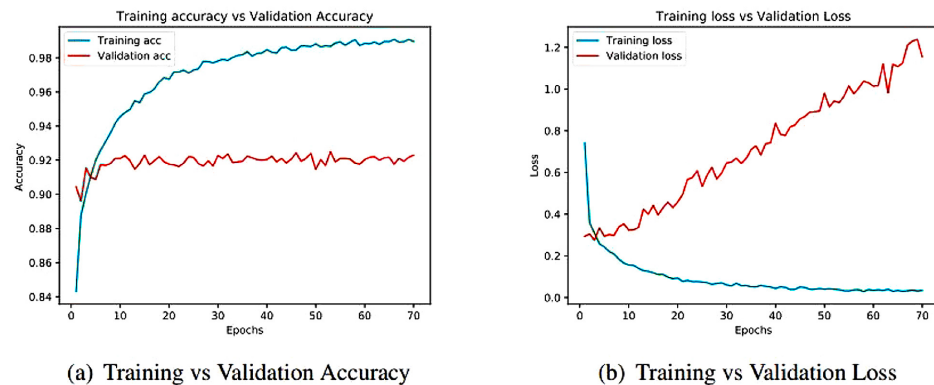



Figure 12. Training and Validation Metrics for Xception

Table 1. Performance Metrics

Model	Accuracy	Precision	Recall	F1-score
InceptionV3	89%	89%	89%	89
ResNet50	37%	32%	37%	29
VGG16	89%	89%	89%	89
VGG19	87%	87%	87%	87
Xception	90%	90%	90%	90
EnsemV3X	91%	91%	91%	91

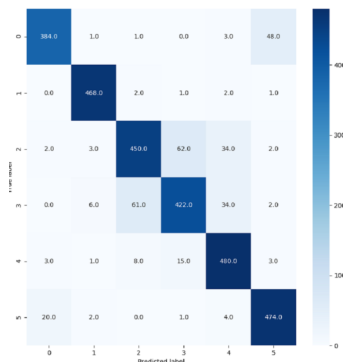



Figure 13. Confusion Matrix for VGG16   **Figure 14.** Confusion Matrix for VGG19

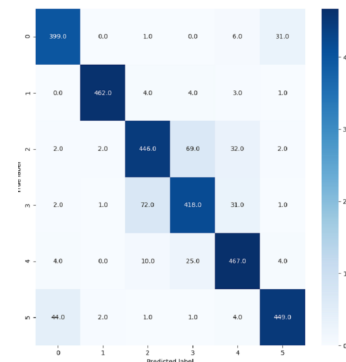
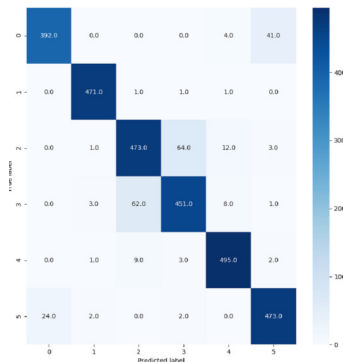
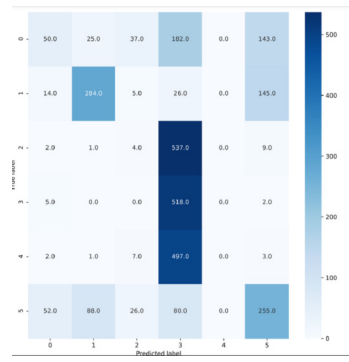



Figure 15. Confusion Matrix for Xception   **Figure 16.** Confusion Matrix for ResNet50

RESULTS

Learning curves (Amari, 1993) obtained after running the models are important sources for measuring the performance of the deep learning model. Training and validation curves are two types of learning curves calculated from training and validation datasets (Xu and Goodacre, 2018) respectively. Loss (Hastie et al., 2009) helps us understand how bad a model performs with every epoch and the extent of its performance in a particular data sample. Meanwhile, accuracy helps denote how accurate the model prediction is to the actual data. Training and the validation curves for each of the architecture are demonstrated. On the one hand, the plot for training accuracy and training loss shows the outcome for the training data. On the other hand, the plot for validation accuracy and loss show the outcome for the validation data. Validation data are used to validate outcomes from the training data and check what the model has learnt from the data. The distance between curves shows the performance of the model.

  Figure 8 illustrates the training versus validation accuracy and loss curve for the InceptionV3 model. Similarly, Figure 9 presents the same curves for the ResNet50 model. VGG16 and VGG19 training versus validation accuracy and loss curves are depicted in Figures 10 and 11, respectively. Figure 12 shows the training versus validation accuracy and loss curve for the Xception model.

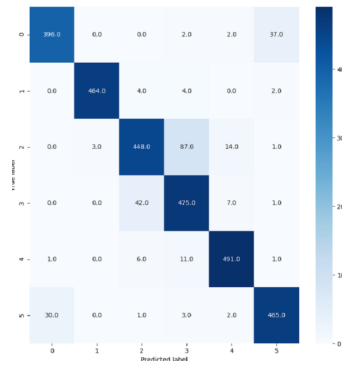
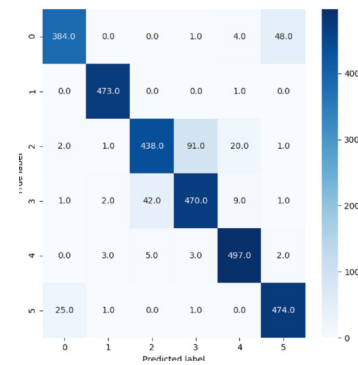



Figure 17. Confusion Matrix for InceptionV3   **Figure 18.** Confusion Matrix for EnsemV3X

DISCUSSION

Accuracy (Hossin and Sulaiman, 2015) is defined as the number of correct predictions made by our model to the total number of predictions. Equation 3 represents the formula for accuracy.

$$A = \frac{1}{n} \sum_{i=1}^{n} \frac{|M_i \cap N_i|}{|M_i \cup N_i|}, \tag{3}$$

where Mi represents the given labels in the dataset, Ni represents the predicted labels in the dataset, and A denotes accuracy.

These terms can be understood in the context of a confusion matrix (Chow et al., 2013). The confusion matrix is a metric that uses four combinations of predicted and actual values. True positive represents true terms that were predicted to be true. True negative is a false term that was predicted to be false. False positive is a term with a negative actual class that was predicted to be positive. False negative is a term with a positive actual class that was predicted to be negative.

VGG16 and VGG19 confusion matrices are shown by Figures 13 and 14, respectively. Figures 15, 16, 17, and 18 depict the confusion matrices for the Xception model, ResNet50 model, InceptionV3 model, and Ensemble models, respectively. These confusion matrices represent the outcome of around 3,000 images used for testing. Similarly, other metrics can also be calculated according to the confusion matrix obtained for each model. The label 0 denotes one of the six classes and the same applies for other labels. The confusion matrix shows how many images were labeled class 0 and actually belonged to class 0 and those that were incorrectly classified. Precision (Hossin and Sulaiman, 2015) and recall (Hossin and Sulaiman, 2015) are expressed as follows:

$$P = \frac{1}{n} \sum_{i=1}^{n} \frac{|M_i \cap N_i|}{|N_i|}, \tag{4}$$

$$R = \frac{1}{n} \sum_{i=1}^{n} \frac{|M_i \cap N_i|}{|M_i|}, \tag{5}$$

where Mi represents the labels given in the dataset and Ni represents the labels predicted for the instance i. Equation 4 depicts the formula for precision and equation 5 depicts the formula for recall. Lastly, equation 6 depicts the equation for f1-score (Hossin and Sulaiman, 2015) can be calculated using precision and recall as follows:

$$F_1 = \frac{1}{n} \sum_{i=1}^{n} \frac{2|M_i \cap N_i|}{|M_i| + |N_i|}. \tag{6}$$

The comparison of performance of all models is presented in Table 1.

CONCLUSIONS AND FUTURE SCOPE

ImageNet, a dense and diverse dataset that has been trained with over 22,000 object categories on the basis of WordNet, has demonstrated excellent performance in object recognition but poor performance in scene recognition. We attempted to perform the task of multilabel scene recognition using ImageNet architectures from the Keras library and Google Colab and then compared their performance, as listed in Table 1. The results showed that ImageNet can achieve satisfactory results and further improve by changing architectures depending on the problem. ImageNet can be used to solve many scene recognition challenges given its very diverse dataset. We also attempted to prepare an ensemble using models previously trained with ImageNet. The InceptionV3 and Xception models obtained the best performance compared with the other models and demonstrated an accuracy of 91% after combining them to form an ensemble.

ImageNet architectures for rich and dense images captured by mobile phone cameras can be the focus of future investigations. Furthermore, ImageNet and its counterpart PlacesCNN can be compared on the same dataset to analyze their performance for both object detection and scene recognition.

REFERENCES

Akiba, T., Suzuki, S., and Fukuda, K. (2017). Extremely Large Minibatch SGD: Training ResNet-50 on ImageNet in 15 Minutes. *arXiv e-prints*, page arXiv:1711.04325.

Akilan, T., Wu, Q. J., Safaei, A., and Jiang, W. (2017). A late fusion approach for harnessing multi-cnn model high-level features. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 566–571. IEEE.

Aly, M. (2005). Survey on multiclass classification methods. *Neural Netw*, 19:1–9.

Amari, S.-I. (1993). A universal theorem on learning curves. *Neural networks*, 6(2):161–166.

Basha, S. S., Dubey, S. R., Pulabaigari, V., and Mukherjee, S. (2020). Impact of fully connected layers on performance of convolutional neural networks for image classification. *Neurocomputing*, 378:112–119.

Bastidas, A. (2017). Tiny imagenet image classification.

Berg, A., Deng, J., and Fei-Fei, L. (2010). Large scale visual recognition challenge (ilsvrc), 2010. *URL http://www. image-net. org/challenges/LSVRC*, 3.

Bloice, M. D., Stocker, C., and Holzinger, A. (2017). Augmentor: An image augmentation library for machine learning. *CoRR*, abs/1708.04680.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258.

Chow, J.-H., Dom, B., and Lin, D.-I. (2013). Confusion matrix for classification systems. US Patent 8,611,675.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., and Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

Furusho, Y. and Ikeda, K. (2019). Resnet and batch-normalization improve data separability. In *Asian Conference on Machine Learning*, pages 94–108.

Gordon, D. F. and Desjardins, M. (1995). Evaluation and selection of biases in machine learning. *Machine learning*, 20(1-2):5–22.

Goyal, S. and Benjamin, P. (2014). Object recognition using deep neural networks: A survey. *CoRR*, abs/1412.3684.

Gulli, A. and Pal, S. (2017). *Deep learning with Keras*. Packt Publishing Ltd.

Han, S., Pool, J., Tran, J., and Dally, W. (2015). Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Herranz, L., Jiang, S., and Li, X. (2016). Scene recognition with cnns: objects, scales and dataset bias. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 571–579.

Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116.

Hossin, M. and Sulaiman, M. (2015). A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2):1.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. (2016). Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer.

Huh, M., Agrawal, P., and Efros, A. A. (2016). What makes imagenet good for transfer learning? *CoRR*, abs/1608.08614.

Intel (2018). *Intel Image classification*. https://www.kaggle.com/puneet6060/intel-image-classification/activity.

Jain, R., Nayyar, A., and Bachhety, S. (2020). Factex: A practical approach to crime detection. In *Data Management, Analytics and Innovation*, pages 503–516. Springer.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678.

Ketkar, N. (2017). Introduction to keras. In *Deep learning with Python*, pages 97–111. Springer.

Khan, A., Sohail, A., Zahoora, U., and Qureshi, A. S. (2019). A survey of the recent architectures of deep convolutional neural networks. *CoRR*, abs/1901.06032.

Khan, S., Rahmani, H., Shah, S. A. A., and Bennamoun, M. (2018). A guide to convolutional neural networks for computer vision. *Synthesis Lectures on Computer Vision*, 8(1):1–207.

Kornblith, S., Shlens, J., and Le, Q. V. (2019). Do better imagenet models transfer better? In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2661–2671.

Koushik, J. (2016). Understanding convolutional neural networks.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Le, Y. and Yang, X. (2015). Tiny imagenet visual recognition challenge. *CS 231N*.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Liu, S. and Tian, G. (2019). An indoor scene classification method for service robot based on cnn feature. *Journal of Robotics*, 2019.

Liu, T., Fang, S., Zhao, Y., Wang, P., and Zhang, J. (2015). Implementation of training convolutional neural networks. *CoRR*, abs/1506.01195.

Lohr, S. (2012). The age of big data. *New York Times*, 11(2012).

Marcelino, P. (2018). Transfer learning from pre-trained models. *Towards Data Science*.

Masud, M. M., Gao, J., Khan, L., Han, J., and Thuraisingham, B. (2008). A practical approach to classify evolving data streams: Training with limited amount of labeled data. In *2008 Eighth IEEE International Conference on Data Mining*, pages 929–934. IEEE.

Mayoraz, E. and Alpaydin, E. (1999). Support vector machines for multi-class classification. In *International Work-Conference on Artificial Neural Networks*, pages 833–842. Springer.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Miller, G. A. (1998). Nouns in wordnet. *WordNet: An electronic lexical database*, pages 23–46.

O'Shea, K. and Nash, R. (2015). An introduction to convolutional neural networks. *CoRR*, abs/1511.08458.

Pan, S. J., Kwok, J. T., and Yang, Q. (2008). Transfer learning via dimensionality reduction. In *AAAI*, volume 8, pages 677–682.

341 Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with
342   region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

343 Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A.,
344   Bernstein, M., C. Berg, A., and Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge.
345   *International journal of computer vision*, 115(3):211–252.

346 Sarkar, D., Bali, R., and Ghosh, T. (2018). *Hands-On Transfer Learning with Python: Implement
347   advanced deep learning and neural network models using TensorFlow and Keras*. Packt Publishing
348   Ltd.

349 Sharif Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf: an
350   astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and
351   pattern recognition workshops*, pages 806–813.

352 Sibi, P., Jones, S. A., and Siddarth, P. (2013). Analysis of different activation functions using back
353   propagation neural networks. *Journal of Theoretical and Applied Information Technology*, 47(3):1264–
354   1268.

355 Simon, M., Rodner, E., and Denzler, J. (2016). Imagenet pre-trained models with batch normalization.
356   *CoRR*, abs/1612.01452.

357 Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image
358   recognition. *CoRR*, abs/1409.1556.

359 Smola, A. J. and Schölkopf, B. (1998). *Learning with kernels*, volume 4. Citeseer.

360 Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a
361   simple way to prevent neural networks from overfitting. *The journal of machine learning research*,
362   15(1):1929–1958.

363 Stone, P. and Veloso, M. (2000). Layered learning. In *European Conference on Machine Learning*, pages
364   369–381. Springer.

365 Sun, L. (2016). Resnet on tiny imagenet. *cs231n.stanford.edu*, 14.

366 Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and
367   Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on
368   computer vision and pattern recognition*, pages 1–9.

369 Torrey, L. and Shavlik, J. (2010). Transfer learning. In *Handbook of research on machine learning
370   applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI Global.

371 Van Der Maaten, L., Postma, E., and Van den Herik, J. (2009). Dimensionality reduction: a comparative.
372   *J Mach Learn Res*, 10(66-71):13.

373 Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. (2018). Deep learning for computer
374   vision: A brief review. *Computational intelligence and neuroscience*, 2018.

375 Wang, L., Guo, S., Huang, W., and Qiao, Y. (2015). Places205-vggnet models for scene recognition.

376 Weiss, K., Khoshgoftaar, T. M., and Wang, D. (2016). A survey of transfer learning. *Journal of Big data*,
377   3(1):9.

378 Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., and Torralba, A. (2010). Sun database: Large-scale scene
379   recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and
380   Pattern Recognition*, pages 3485–3492. IEEE.

381 Xu, B., Wang, N., Chen, T., and Li, M. (2015). Empirical evaluation of rectified activations in convolutional
382   network. *CoRR*, abs/1505.00853.

383 Xu, Y. and Goodacre, R. (2018). On splitting training and validation set: A comparative study of
384   cross-validation, bootstrap and systematic sampling for estimating the generalization performance of
385   supervised learning. *Journal of Analysis and Testing*, 2(3):249–262.

386 Yao, L. and Miller, J. (2015). Tiny imagenet classification with convolutional neural networks. *CS 231N*,
387   2(5):8.

388 Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural
389   networks? In *Advances in neural information processing systems*, pages 3320–3328.

390 Zoph, B., Ghiasi, G., Lin, T.-Y., Cui, Y., Liu, H., Cubuk, E. D., and Le, Q. V. (2020). Rethinking
391   pre-training and self-training.