Inductive and Transductive Transfer Learning: COMODEL

ADITYA PAPRUNIA[1], JYOTI MALHOTRA[2]
[1, 2] *Department of CSE, MIT School Of Engineering, MIT ADT University, Pune, Maharashtra*

Abstract- Transfer Learning is a commonly used method in deep learning to train neural networks from a previously trained model. It focuses on using knowledge gained while solving one problem to solve a similar problem without explicitly developing a solution for the new problem. Similarly, previously trained models were created for a specific goal, but their learnings can be applied to other tasks as well. The new model uses the pre-trained model as the starting point for the new task. It extracts meaningful features from previous models and uses them for the current problem. In this research paper we compare the performance of the different pre-trained models; as per their accuracy with training data and validation data to get an idea of which model is best suited to give better results when transfer learning is used.

I. INTRODUCTION

The use of transfer learning has allowed developers and researchers to solve complex problems using much fewer resources and achieve better results in a fraction of the time [1]. It is often used in image classification problems to train new models as datasets required to train transfer-learning models are much smaller than regular datasets.

The imagenet dataset, which contains roughly 1.4 mil lion photos belonging to 1000 different classes, was u sed to train the previously learned models. It contains annotated photographs of different objects and was developed specifically for computer vision problems by a group of researchers at Stanford University. The models used for this paper were namely ResNet50, InceptionV3, and VGG19. All the models used the weights from the image net dataset. ResNet50 is a 50-layer deep neural network created in 2015 and was introduced in a research paper called “Deep Residual Learning for Image Recognition” [7]. InceptionV3 is a 48-layer deep neural network based on the paper "Rethinking the Inception Architecture for Computer Vision" in 2014. VGG19 is a 19-layer deep neural network created in 2014. The above models are the most popularly used image classification models and are known to give excellent results.

This manuscript performed the analysis on three datasets. The first dataset contains images of cats and dogs. The second contained the images of hyenas and cheetahs, which is similar to the first dataset of cats and dogs. Moreover, the last one contained images of helicopters and airplanes, which is quite different from the first dataset.

Previous research work has mainly focused on using imagenet weights to compare the performance of different models but custom weights have not been used to compare their performance when used with a similar dataset (inductive) and with a very different dataset (transductive). Thus, we analyzed and assessed the difference between the two and compared the performance of the different models.

II. LITERATURE REVIEW

This section highlights the work performed by various authors. There are many projects conducted on research learning, like an article from towardsdatasceince.com by Gabriel Cassimiro[2]. The project uses VGG16 for transfer learning. It gives us a brief overview of transfer learning. It uses the TensorFlow flower dataset for training but does compare the performance of VGG16 with other models. It also does not include the training time and resources used while training the model.

The authors of the paper “A comparative study of methods for transductive transfer learning” [3] have highlighted the importance of transfer learning and have compared different types of models like SVMs with their technique.

The paper "A general framework for scalable transductive transfer learning" [4] focuses on a general

approach to transductive transfer learning by comparing the input features of the first dataset with the features of the second dataset which is related. The authors have used abstract transductive transfer learning in the paper i.e. only labeled data is available in the original dataset and only unlabelled data is available in the second, related dataset

The paper "Transfer learning for pedestrian detection" [5] uses unseen examples from the original dataset and adds them to the new dataset and also uses a new type of classification model. This is an example of inductive transfer learning but it is unsupervised.

In the research paper titled "A review on transfer learning in EEG signal analysis"[6] the authors have used transfer learning to train models using less EEG data as transfer learning requires less data as compared to traditional model training.

However, all the above papers lack a comparative study between the models and their performance on different datasets.

III. WORKING ON COMODEL

This section covers the details of COMODEL i.e. COMparison mODEL of transfer learning. The below diagram (Figure 1) is a general representation of the COMODEL:

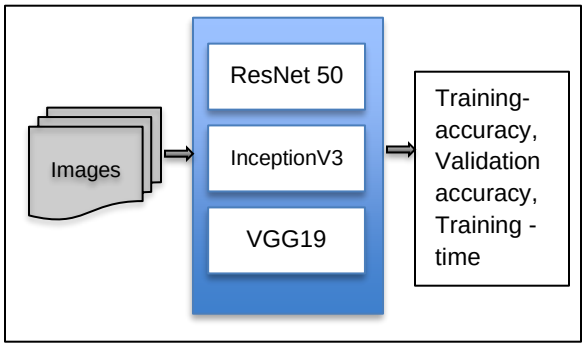

Figure 1: COMODEL

Three pre-trained models were chosen for the comparison which are ResNet50, InceptionV3, and VGG19. Each model used weights from imagenet. Initially, for the first dataset containing images of cats and dogs, none of the layers were frozen (i.e. all layers were trainable) in each model. After the training was complete the weights from the weights obtained from the training were used with a similar dataset containing images of cheetahs and hyenas (inductive). The same weights from the original dataset were used with a random dataset containing images of helicopters and airplanes. The results are recorded for the same and are discussed further.

a. ResNet50

The architecture of ResNet50 consists of 50 Convolution layers along with 1 MaxPool and 1 Average Pool layer (refer to Figure 2) -

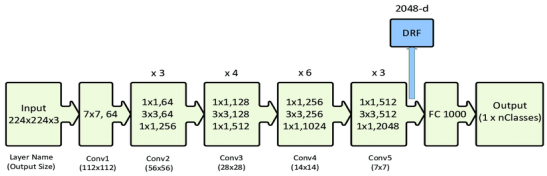

Figure 2: ResNet50 Architecture

Firstly, ResNet50 dataset was trained with the original dataset containing images of cats and dogs along with the following training parameters-

Dataset - Cats vs Dogs
Number of Training Images - 2520
Number of validation Images - 252
Image shape - 224,224
Weights - imagenet
Layers trainable - True
Batch Size - 8
Epochs - 15
Optimizer (learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 94% and a validation accuracy of 96% (refer to Figure 3). The weights obtained from this training were saved to be used for training with the next model(resnet50_original_weights.h5).

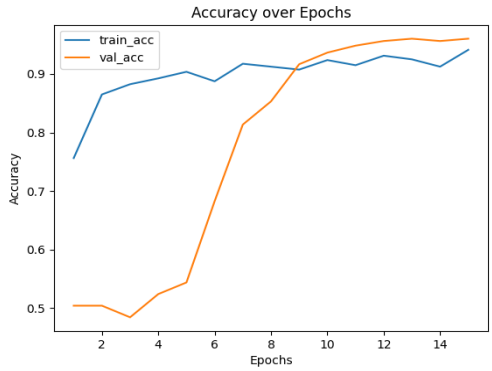

Figure 3: ResNet50 train_acc vs val_acc for Cats vs Dogs dataset

Next, the ResNet50 model was trained on a similar dataset to the original dataset containing images of cheetahs and hyenas. The following parameters were used during training-

Dataset - Cheetah vs Hyena
Number of Training Images - 1600
Number of validation Images - 100
Image shape - 224,224
Weights - resnet50_original_weights.h5 (weights from previous training)
Layers trainable - False
Batch Size - 8
Epochs - 15
Optimizer(learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 98.87% and validation accuracy of 99.50% (Figure 4)

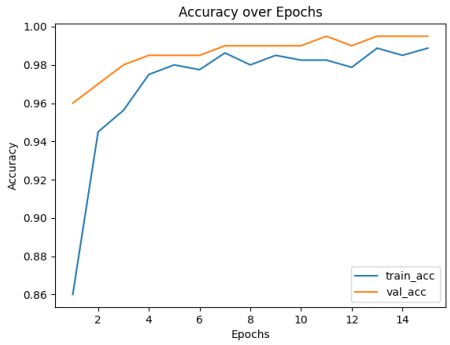

Figure 4: ResNet50 train_acc vs val_acc for Cheetahs vs Hyenas dataset

The final training was performed on a dataset that was unrelated to the original dataset containing pictures of helicopters and airplanes. The following parameters were used for the training:

Dataset - Helicopters vs Airplanes
Number of Training Images - 2520
Number of validation Images - 252
Image shape - 224,224
Weights - resnet50_original_weights.h5 (weights from previous training)
Layers trainable - False
Batch Size - 8
Epochs - 15
Optimizer(learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 91.60% and validation accuracy of 91.70%(as shown in Figure 5)

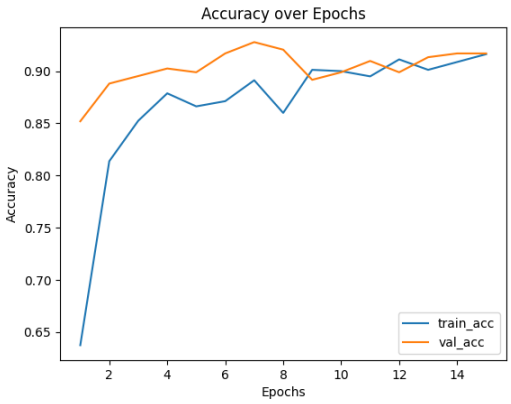

Figure 5: ResNet50 train_acc vs val_acc for Helicopters vs Airplanes dataset

Observations - From the above results, it is observed that the accuracy of the dataset similar to the original dataset is much better than the accuracy of the dataset, which is unlike the original dataset for both training and validation images. The training time for ResNet50 was more than the other two models but the training time for the datasets using weights from the original dataset is much lesser and the resources used were significantly less.

Time required to train one epoch in the original dataset (Cats vs Dogs) - 252s (average)
Time required to train one epoch in the similar dataset (Cheetah vs Hyenas) and random dataset (Helicopters vs Airplanes) - 99s (average)

b. InceptionV3

The architecture of InceptionV3 consists of 48 layers (Figure 6)

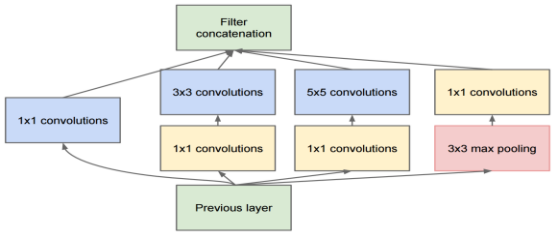

Figure 6: InceptionV3 Architecture

The Softmax layer was replaced with a Sigmoid layer for binary classification.

Firstly InceptionV3 was trained with the original dataset containing images of cats and dogs. The following were the parameters used for training-

Dataset - Cats vs Dogs
Number of Training Images - 2520
Number of validation Images - 252
Image shape - 299,299
Weights - imagenet
Layers trainable - True
Batch Size - 8
Epochs - 15
Optimizer (learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 92% and a validation accuracy of 95.60% (Figure 7). he weights obtained from this training were saved to be used for training with the next model(inceptionv3_original_weights.h5).

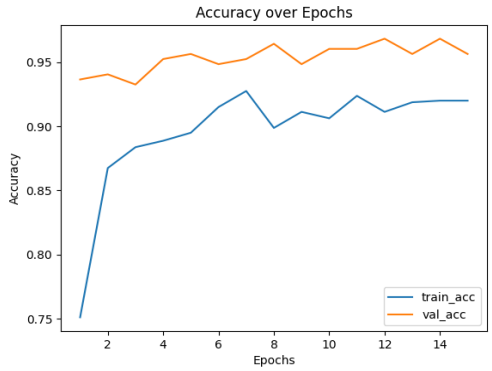

Figure 7: InceptionV3 train_acc vs val_acc for Cats vs Dogs dataset

Next, the InceptionV3 model was trained on a similar dataset to the original dataset containing images of cheetahs and hyenas. The following parameters were used during training-

Dataset - Cheetah vs Hyena
Number of Training Images - 1600
Number of validation Images - 100
Image shape - 299,299
Weights - inceptionv3_original_weights.h5 (weights from previous training)
Layers trainable - False
Batch Size - 8
Epochs - 15
Optimizer(learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 99.60% and a validation accuracy of 99.50% (refer to Figure 8).

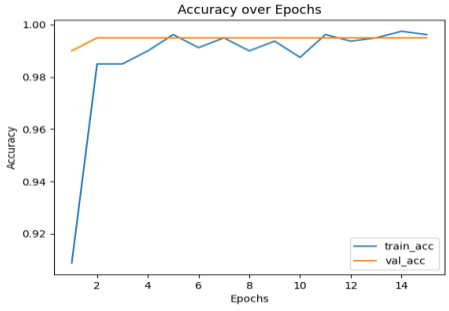

Figure 8: InceptionV3 train_acc vs val_acc for Cheetahs vs Hyenas dataset

The final training was performed on a dataset that was unrelated to the original dataset containing pictures of helicopters and airplanes. The following parameters were used for the training:

Dataset - Helicopters vs Airplanes
Number of Training Images - 2520
Number of validation Images - 252
Image shape - 299,299
Weights - inceptionv3_original_weights.h5 (weights from previous training)
Layers trainable - False
Batch Size - 8
Epochs - 15
Optimizer(learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 90.50% and validation accuracy of 90.60% (Figure 9)

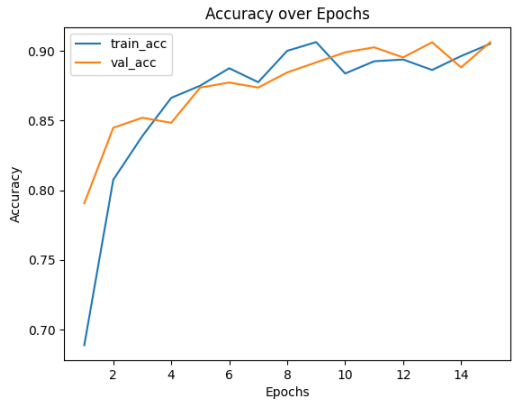

Figure 9: InceptionV3 train_acc vs val_acc for Helicopters vs Airplanes dataset

Observations- InceptionV3, like ResNet50, achieved better accuracy with a similar dataset than with an unrelated dataset, and the training time is significantly reduced when weights from the original dataset were employed. The original dataset took less time to train than ResNet50 and VGG19.

Time taken to train the original dataset (Cats vs Dogs) - 220s (average)

Time required to train one epoch in the similar dataset (Cheetah vs Hyenas) and random dataset (Helicopters vs Airplanes) - 84s (average)

c. VGG19

VGG19 consists of 19 layers that include 16 convolution layers, 3 fully connected layers, 5 MaxPool layers, and 1 SoftMax layer. In our model, the softmax layer is replaced with the sigmoid layer for binary classification. (Figure 10)

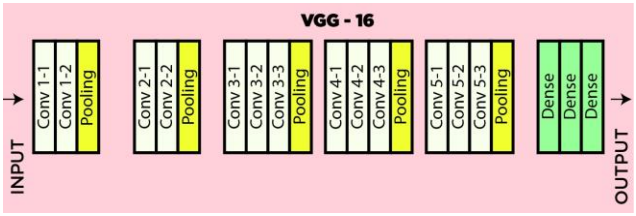

Figure 10: VGG19 architecture

Firstly, the VGG19 model was trained with the original dataset containing images of cats and dogs. The following were the parameters used for training-

Dataset - Cats vs Dogs
Number of Training Images - 2520
Number of validation Images - 252
Image shape - 224,224
Weights - imagenet
Layers trainable - True
Batch Size - 8
Epochs - 15
Optimizer (learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 75.80% and a validation accuracy of 80.50% (Figure 11). he weights obtained from this training were saved to be used for training with the next model (vgg19_original_weights.h5).

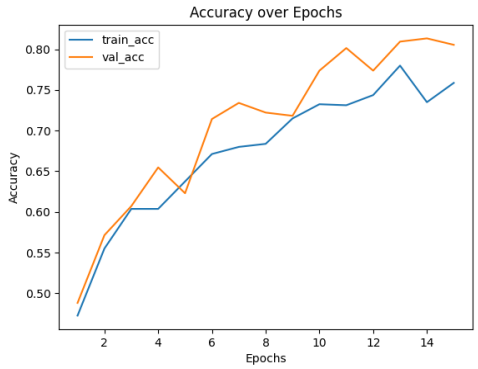

Figure 11: VGG19 train_acc vs val_acc for Cats vs Dogs dataset

Next, the VGG19 model was trained on a similar dataset to the original dataset containing images of cheetahs and hyenas. The following parameters were used during training-

Dataset - Cheetah vs Hyena
Number of Training Images - 1600
Number of validation Images - 100
Image shape - 224,224
Weights - vgg19_original_weights.h5 (weights from previous training)
Layers trainable - False
Batch Size - 8
Epochs - 15
Optimizer (learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 95.25% and a validation accuracy of 97.50% (Figure 12).

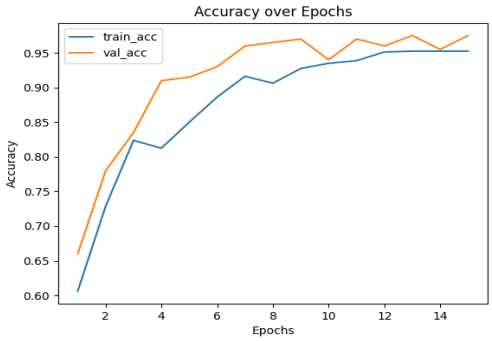

Figure 12: VGG19 train_acc vs val_acc for Cheetah vs Hyenas dataset

The final training was performed on a dataset that was unrelated to the original dataset containing pictures of helicopters and airplanes. The parameters used for the training are mentioned below :

Dataset - Helicopters vs Airplanes
Number of Training Images - 2520
Number of validation Images - 252
Image shape - 224,224
Weights - vgg19_original_weights.h5 (weights from previous training)
Layers trainable - False
Batch Size - 8
Epochs - 15
Optimizer(learning rate) - Stochastic Gradient Descent (0.0001)
Loss function - binary_crossentropy
Activation Function - Sigmoid

After training, the model gave a training accuracy of 85.50% and a validation accuracy of 90.25% (Figure 13).

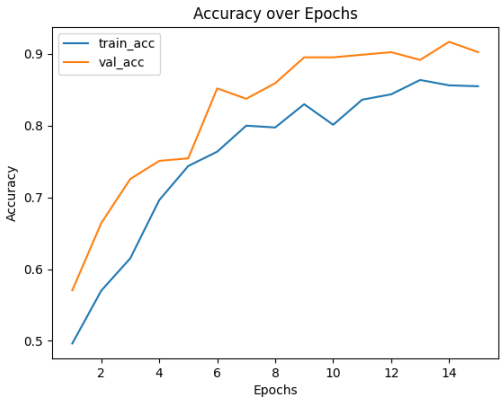

Figure 13: VGG19 train_acc vs val_acc for Helicopters vs Airplanes dataset

Observations- The VGG19 model gave lesser accuracy than both the other models for all datasets and the training time required for it was also quite high.

Time taken to train original dataset (Cats vs Dogs)- 319s (average)
Time required to train one epoch in the similar dataset (Cheetah vs Hyenas) and random dataset (Helicopters vs Airplanes)- 149s (average)

IV. CONCLUSION

The comparison shows that ResNet50 has the best performance of all the three models according to accuracy. The accuracy of InceptionV3 was quite close but not as good as ResNet50. VGG19 gave the least accuracy of all the three models. A possible explanation for this could be the presence of more layers in ResNet50 (50 layers) which means more trainable parameters. This is also consistent with our other results as InceptionV3 with 48 layers and nearly 2 million more trainable parameters than VGG19 gave a better result. The training time for InceptionV3 was the least followed by ResNet50 and lastly, VGG19.

Figure 14 visualizes the final results of all models with all 3 datasets for val_acc.
Table I summarizes the COMODEL observations.

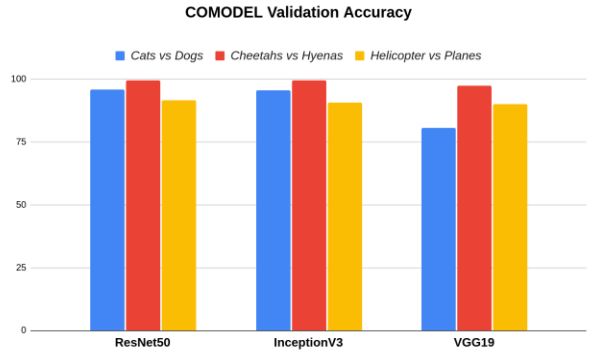

Figure 14: Comparison of Validation accuracy of all models with all 3 datasets

Table I: COMODEL observations

Model Name	Image shape	val_acc	Training Time
Cats vs Dogs (Train Images - 2520, Val. Images - 252)			
ResNet50	224, 224	96%	252
InceptionV3	299, 299	95.60%	99
VGG19	224, 244	80.50%	99
Cheetahs vs Hyenas (Train Images - 1600, Val. Images - 100)			
ResNet50	224, 224	99.50%	220
InceptionV3	299, 299	99.50%	84
VGG19	224, 224	97.50%	84
Helicopters vs Planes (Train Images - 2520, Val. Images - 252)			
ResNet50	224, 224	91.70%	319
InceptionV3	299, 299	90.60%	149
VGG19	224, 224	90.25%	149

V. ACKNOWLEDGEMENT

I would like to thank my professor and guide Dr. Jyoti Malhotra for her guidance and constant support. I would also like to thank the HoD of the Computer Science department at MIT ADT University Prof. Shraddha Phansalkar and the Dean of Engineering Dr.Rajni Sachdeo for their mentorship throughout the research period.

REFERENCES

[1] Mustafa, B., Loh, A., Freyberg, J., MacWilliams, P., Wilson, M., McKinney, S. M., ... & Natarajan, V. (2021). Supervised transfer learning at scale for medical imaging. *arXiv preprint arXiv:2101.05913*.

[2] Article in towardsdatasceince.com by Gabriel Cassimiro- https://towardsdatascience.com/transfer-learning-with-vgg16-and-keras-50ea161580b4

[3] Arnold, A., Nallapati, R., & Cohen, W. W. (2007, October). A comparative study of methods for transductive transfer learning. In Seventh IEEE international conference on data mining workshops (ICDMW 2007) (pp. 77-82). IEEE.

[4] Bahadori, M. T., Liu, Y., & Zhang, D. (2014). A general framework for scalable transductive transfer learning. Knowledge and information systems, 38(1), 61-83.

[5] Cao, X., Wang, Z., Yan, P., & Li, X. (2013). Transfer learning for pedestrian detection. *Neurocomputing*, *100*, 51-57.

[6] Wan, Z., Yang, R., Huang, M., Zeng, N., & Liu, X. (2021). A review on transfer learning in EEG signal analysis. *Neurocomputing*, *421*, 1-14.

[7] He, Kaiming & Zhang, Xiangyu & Ren, Shaoqing & Sun, Jian. (2016). Deep Residual Learning for Image Recognition. 770-778. 10.1109/CVPR.2016.90.