**Architectures for Transfer Learning on Various Resolutions**

**Anya Mikaela Greene 1, David Jose Florez Rodriguez2,3,**

1 Palo Alto High School, Stanford, CA, USA

2 Papaya Institute, Phoenix, Arizona, USA

3 Electrical Engineering, Stanford University, Stanford, CA, USA

**Student Authors**

Anya Mikaela Greene, High School

**SUMMARY** *250 words or less, one paragraph*

The larger problem lies within the lack of accessibility for AI models. They are expensive and time-consuming, making them less accessible for everyday users to learn from them. The larger question is ‘how can we make AI accessible with existing models?’ Transfer learning can leverage large datasets and massive computing power from the best AI labs to help smaller teams looking at AI applications6,7. The hypothesis is was that more complex models, including more diverse layers, built with transfer learning will outperform simpler models also made from transfer learning. We used the VGG161 model and 3 separate datasets spanning textures (MINC), satellite views (EUROSAT), and standard classification images (CIFAR10). We tested multiple transfer learning model architectures. In general, we found that simpler architectures are more accurate and reliable than more complicated ones. This is very accessible, as more complex architectures can be hard to buid and tune. These findings support that transfer learning is best with the simplest attached models on top of the transferred, pre-trained architecture.

**INTRODUCTION**

The overarching scientific topic of the manuscript is transfer learning. Transfer learning is the process in which an already existing model is reused and trained on another dataset or for another purpose. It often beats other methods of AI based learning and can often more effective than creating a model from scratch because the model has already been used and tested6,7. Making AI more accessible by reducing the costs and making it less time consuming opens up the field of AI technology, so that more people can understand and learn from the resources available. This enhances the machine learning field and could help users detect and solve problems using AI. While there are many opinions surrounding the use and misuse of AI as it is becoming increasingly popular and common, it is clear that AI will play a large role in the future of our society. That being said, it is important that more people understand it in more depth, and that it is more accessible for people to understand its structure and applications, rather than just use or not use programs for personal use. This work tests transfer learning on three datasets. The transferred model is VGG16, which has incredible accuracy on the ImageNet dataset spanning 1000 visual categories8. Various architectures are built after the convolutional layers of VGG16, testing to see which architecture will best perform on three different datasets. The architectures, defined in We hypothesize that the larger architectures with more layer diversity will have higher validation accuracy than smaller architectures, as larger architectures can learn more and thus ‘transfer’ better.

**RESULTS**

The linear models did best in all cases. This is clear in Figure 1, Figure 2, and Figure 3. They outperformed models in which we added dropout, batchnorm, and dense layers. The validation accuracies are lower than the training accuracies in all cases. For MINC3, the linear model was accurate up to 99% of the time on the training dataset, and was accurate 46% of the time on the validation dataset. The MINC learning curves are in Figure 2. For CIFAR104,5, the model was accurate up to 99% of the time as well, and the model was 67% accurate on the validation dataset. Figure 3 displays the CIFAR10 learning curves. For the EuroSat2 data, the model was accurate up to 96% on the training data, and was accurate 88% of the time on validation data. Figure 1 shows the EuroSat learning curves.

The next best performance comes from the second smallest architecture on the EuroSat data. The EuroSat ‘nothing’ model had a validation accuracy of 60% and training accuracy was 53%.

The performance of all other models were about the same, as they did not exceed 20% accuracy, and did not perform well enough to be of much interest.

**DISCUSSION**

The results do not support the hypothesis of larger architectures working best with transfer learning models. The opposite of our hypothesis was seen. The smallest models, linear models for all three datasets, outperformed all other models. The validation accuracies are below the training accuracies in all cases, which is expected, as models are not expected to perform as well on unseen data in the validation set as they do on the training data which supports the model fitting. The linear models achieved near perfect training accuracies and strong validation accuracies much better than random-chance performance. Our data could have also been overfitting in the MINC and CIFAR datasets, as the gaps between validation and training performances were large.

These models also might not have performed as well as they could have because there was not enough training data from them to learn from. All models could have benefitted from having more data and from having more epochs as well. We were limited by the materials at our disposal, as we used the free version of Google Colab, which only has 12 GBs of RAM and 100 GBs of storage. We were also limited by computation times, because we were running the models on portable computers connected to Google Colab servers and not other ML optimized computers with dedicated servers, the models took longer to run and complete. An approximate maximum run-time of about 6 hours was also present on our version of Google Colab.

While all models and sets of data could have benefitted from having more epochs, one thing that could have also affected the accuracy for CIFAR10 are the original image dimensions. When resizing the images, for MINC we shrunk the pictures to match the VGG16, 224-pixel width and height. So the resolution of these images was decreased, but still as good as what VGG16 was trained on. But for Cifar, we increased the sizes of the images, making them more pixelated and harder to categorize. Increasing the size of these low-resolution images doesn’t increase the information density in their pixels. The Cifar images were originally 32x32 pixels, and were resized to 224-pixel dimensions, and had the least resolution among the three datasets that we used. MINC had the highest resolution, with thousands of pixels by thousands of pixels. EuroSat was originally 64x64 pixels, also having a lower resolution than the 224X224. Nevertheless, EuroSat models performed the best, so it is not clear what effect original resolution has on model performance.

Finally, the various visual patterns present in the images vary greatly between the three datasets. These differences can be seen in Figure 4. CIFAR10 images are pixelated and largely rely on rough shapes with limited contrast or hard edges. One might assume these characteristics make learning difficult. Indeed, CIFAR10’s best results are the worst of the various datasets’ best models. EuroSat images, from satellites, vary greatly in color and have more defined edges and textures than CIFAR10, but are still somewhat pixelated when compared to a proper 224X224 pixel image. It is possible that the color variety across pictures could assist a model in classification. The MINC dataset, showcasing various construction and natural materials from various photography sources, have the best resolution in our research and include ample background colors and objects that could distract a model from learning.

**MATERIALS AND METHODS**

**Materials:**

The materials that we used included: a computer, Google Colab, google Drive, Github, the pandas library, PIL library, NumPy library, and TensorFlow library, as well as the datasets (CIFAR10 with 10 categories, EuroSat with 7 categories, and MINC with 17 categories).

We did all the coding on Google Colab, and made copies of all code files in order to build upon previous work and have records of past testing and analysis.

**Methods:**

We started by downloading all the datasets and moving them to a google drive folder as png or jpg files. Once they were all in the same place and more easily accessible, we had to resize the images to 224x224 pixels, to all be the same size as those first used by the VGG16 model which is the base for our transfer learning. Once resized with the PIL library, we converted the images into arrays and made their labels into one hot vectors in order to identify the categories in which they belonged to. One-hot vectors are vectors of mostly zeroes and a single one in the place matching the category number.

We then loaded the VGG16 model through Keras in our notebook, but did not import the last couple of layers from the pre-trained model. Because VGG16 was reliable when tested with 1000 categories, we assume that it would be accurate when tested with smaller numbers of categories (ranging from 7 to 17). The internal layers of the model identify and separate parts of the images in order to detect what the images are and categorize them. Once removing the last layers, we modified the model into four separate machine learning models and one linear model; one called ‘nothing’, with only 2 dene layers added, one called ‘everything’, which was like the ‘nothing’ model but with dropout and batch norm layers, one called ‘batchnorm’, with dense layers and batchnorm layers, and one called ‘dropout’, with dense and dropout layers.

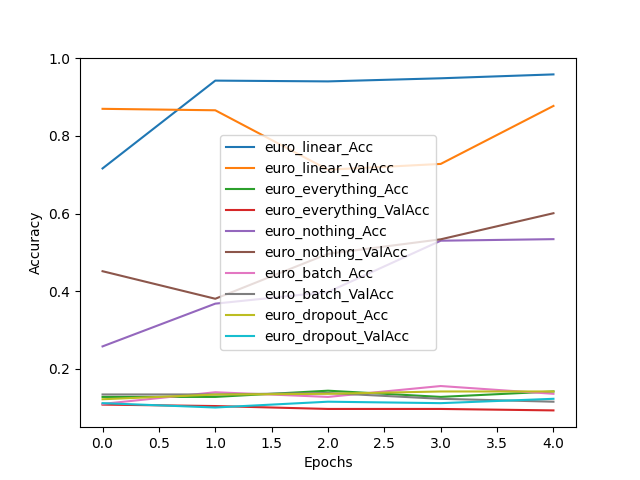
We made sure to shuffle the samples so that they were not in the order that we loaded them into the code. After shuffling the order, we separated the samples and took a portion of it to be validation data. We did not use 15% of the loaded data in training, and instead used that reserved data to validate the accuracy of our model after it was done training on the other 85% of the data. There were 500 images for training each model.

We tested all four models with each of the three datasets, to find out which model performed best and which dataset was best suited for our purposes. Each model was run for 5 epochs and took around 30 minutes to complete.

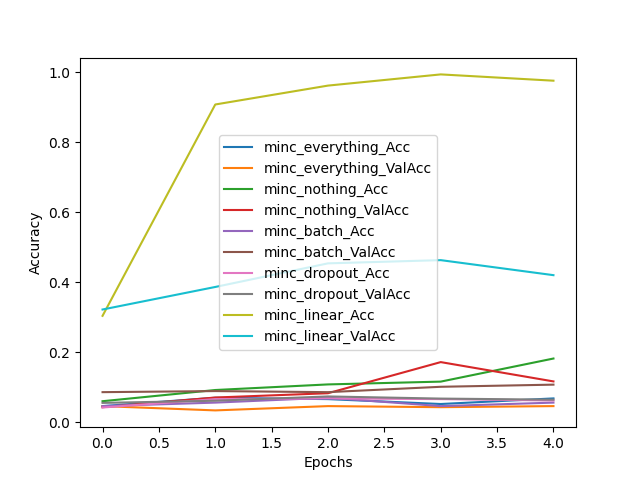
All the code is available in our Github9.

**REFERENCES**

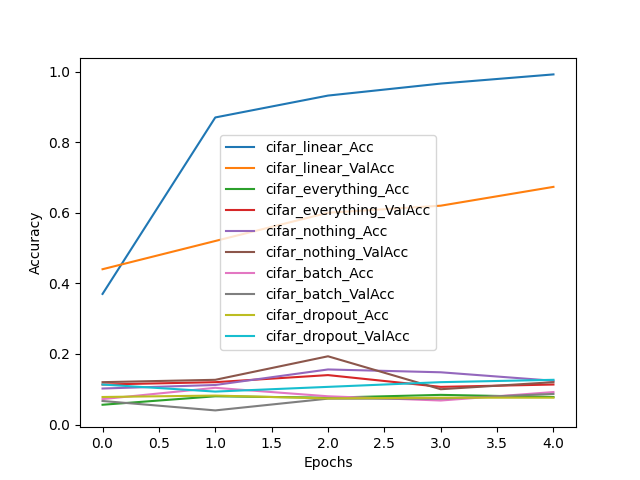
1. Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
2. P. Helber, B. Bischke, A. Dengel and D. Borth, "EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 12, no. 7, pp. 2217-2226, July 2019, doi: 10.1109/JSTARS.2019.2918242.
3. Bell, Sean, et al. "Material Recognition in the Wild with the Materials in Context Database." Papers with Code, paperswithcode.com/paper/material-recognition-in-the-wild-with-the. Accessed 14 Oct. 2023.
4. Krizhevsky, Alex, et al. "The CIFAR-10 Dataset." Alex Krizhevsky, www.cs.toronto.edu/~kriz/cifar.html. Accessed 14 Oct. 2023.
5. Krizhevsky, Alex. "Learning Multiple Layers of Features from Tiny Images." Department of Computer Science, University of Toronto, 8 Apr. 2009, www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf. Accessed 14 Oct. 2023.
6. Zhuang, Fuzhen, et al. "A Comprehensive Survey on Transfer Learning." arXiv, https://doi.org/10.48550/ARXIV.1911.02685.
7. Hosna, Asmaul et al. “Transfer learning: a friendly introduction.” Journal of big data vol. 9,1 (2022): 102. doi:10.1186/s40537-022-00652-w
8. "Keras Applications." Review of Keras. Keras, keras.io/api/applications/. Accessed 14 Oct. 2023.
9. Greene, Anya M., and Florez Rodriguez, David Jose. "Texture2Forest-TransferLearning." 2023. Github. [github.com/ThePapayaInstitute/Texture2Forest-TransferLearning](https://github.com/ThePapayaInstitute/Texture2Forest-TransferLearning)

******

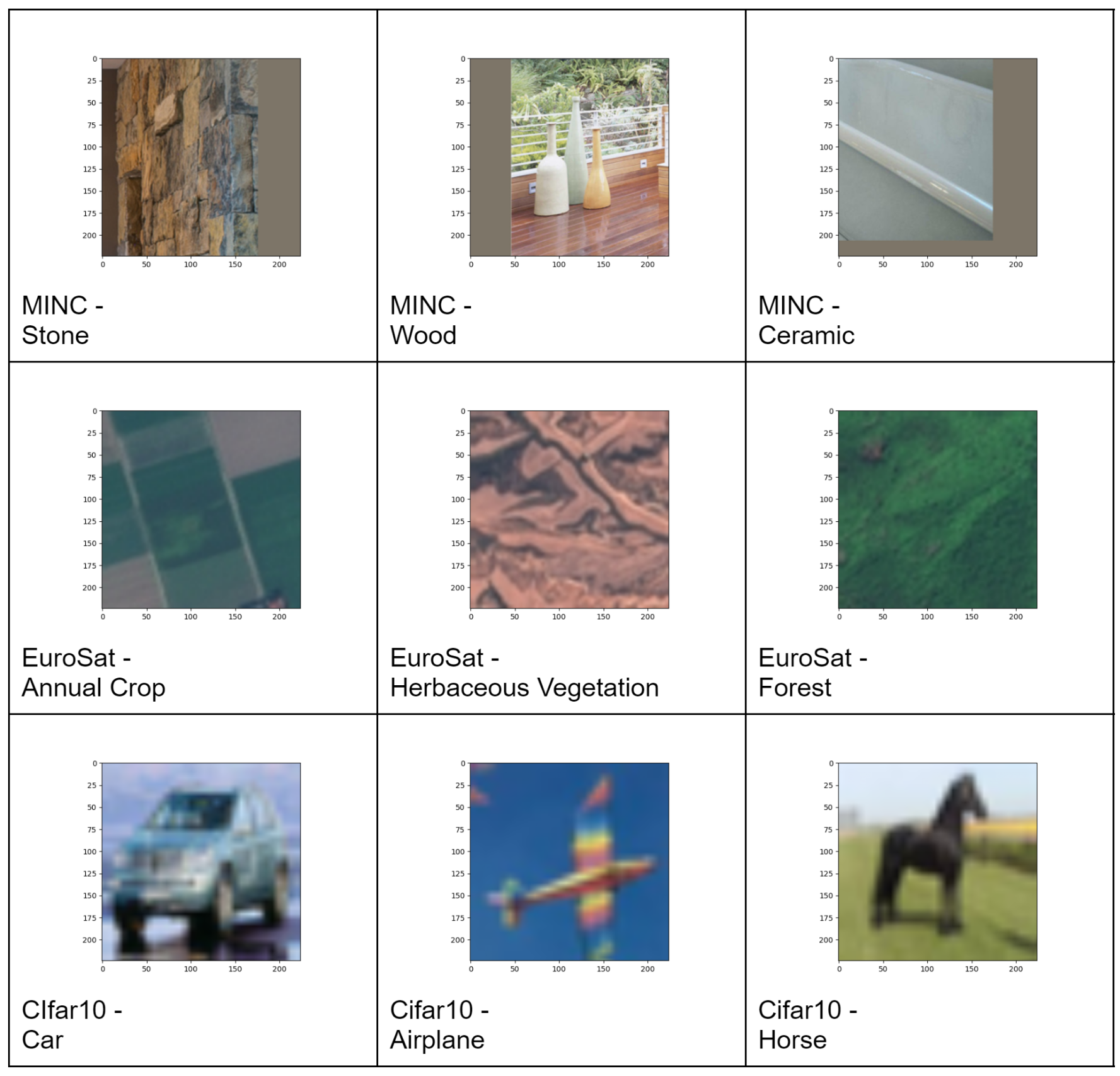
**Figure 1. Training and Validation Accuracy on the EuroSat database.** The linear model had the best results, followed by the “nothing” model which had only dense layers between the VGG16 output and the EuroSat categories prediction. Most other models hardly beat random chance.

******

**Figure 2. Training and Validation Accuracy on the MINC database.** The linear model had the best results. The difference between the final training accuracy and the validation accuracy in this dataset was the largest seen in this work, over 50 percentage points large. All other models hardly reached 20% accuracy.

******

**Figure 3. Training and Validation Accuracy on the CIFAR database.** The linear model had the best results. The difference between the validation and training accuracy here was generous, at nearly 20 percentage points. No other models approached a performance that came close to competing.



**Figure 4. Sample images from the three datasets.** The resolution is highest in the MINC images, up to thousands by thousands of pixels, and this is visibly clearer the CIFAR10 images, which were originally 32 by 32 pixels. The EuroSat images were 64 by 64 pixels originally.