# Vision Transformer Image Transfer Learning Report

Annabel Truong
*anptruon@ucsc.edu*
*University of California, Santa Cruz*

Richard Dao
*rqdao@ucsc.edu*
*University of California, Santa Cruz*

## 1 Base Model

For our transfer learning project, after trying out multiple base models, including ConvNeXt, EfficientNet, ResNet-50, and RegNet-32/128, we came to the decision to use a Vision Transformer.

With ViT_L_16, we were able to achieve about a 81% test accuracy on Kaggle using Cross Entropy Loss and Stochasic Gradient Descent with a learning rate of 0.001 and a momentum value of 0.9.

## 2 Environment

For simplicity, we did the majority of our testing on Google Colab with some bits of the training runs being done simultaneously on our local computers using personal GPUs.

### 2.1 Python Libraries Required

We opted to use PyTorch instead of Keras & Tensorflow for our project as TensorFlow required us to run an older version of Python in order for it to utilize the GPU. In order to use our model, we imported the following libraries:

```
torchvision
torch
numpy
PIL
glob
sklearn
```

We also import modules from aforementioned libraries as well as libraries that help us visualize our data, such as:

```
torchinfo
seaborn
random
matplotlib
```

Each of these libraries and modules contributed to loading, training, augmenting, and testing our data.

## 3 Usage

### 3.1 Where to Find Everything

A link to the Google Drive folder where everything is contained can be found here containing all of our files including: our training environment (Google Colab Python Notebook), train/test files, submission.csv files, our parameter weight files (.pth files), and our final presentation.

### 3.2 How to Run Our Model

Step by step instructions on how our model was trained can be found in the Colab Python Notebook file.

For convenience, if you only want to run our pre-trained weights, you just need to run all the cells in Section 6 rather than the whole document. Be sure to specify whether you want the regularly trained model or the finetuned model by replacing the *file_name* variable to their respective .pth files. See Section 4 Parameter Weights of this document for more details.

Please ensure that you are utilizing a GPU or the code will throw you an error at all instances where there is a *.cuda()* call.

## 4 Parameter Weights

Our final submission to Kaggle is our 100 epoch run of our ViT_L_16 model using weights:

```
ViT_L_16_Weights.IMAGENET1K_SWAG_E2E_V1
```

As mentioned in Section 3 Usage, our saved and transfer-trained model weights can be found in our Google Drive folder here. We have two .pth submissions, one with just the head trained and one with one encoder layer unfrozen and trained. They are respectively named:

```
vit_l_16_100epochs.pth
vit_l_16_100epochs_ft.pth
```

## 5 Contributions

### 5.1 Annabel's Contributions

Annabel's contributions to the project included being in charge of handling the data, defining the fine-tuning loop, and testing out ConvNext, EfficientNet, and ResNet. Annabel imported the data to the Colab file, split the data into train/validation sets, and explored and applied different data augmentations for the training set. In the Colab file, Annabel's contributions include all of Section 1, Section 2 and Section 5.

### 5.2 Richard's Contributions

Richard's contributions to the project included being in charge of defining the training loop and testing loop so that models predictions could be submitted to Kaggle. Richard trained and tested 3 of the models including RegNet-128, RegNet-32, and Vision Transformer. Richard's contributions in the Colab file include all of Section 3, Section 4, and Section 6.

### 5.3 Mutual Contributions

The final presentation and report responsibilities were shared.

## References

[1] W. Wang, J. Yu, Z. Liu, and Y. Yang, "Translating languages with generative adversarial networks: A survey," *arXiv preprint arXiv:2010.11929*, 2020. [Online]. Available: https://arxiv.org/abs/2010.11929

[2] Y. Zhang, Y. Mao, H. Liu, and D. Huang, "A survey on deep learning for named entity recognition," *arXiv preprint arXiv:2104.00298*, 2021. [Online]. Available: https://arxiv.org/abs/2104.00298

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[4] S. Liang, Y. Pan, T. Darrell, and J. Gao, "Transformer-based few-shot learning: A survey," *arXiv preprint arXiv:2201.03545*, 2022. [Online]. Available: https://arxiv.org/abs/2201.03545

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez,  Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:2003.13678*, 2017. [Online]. Available: https://arxiv.org/abs/2003.13678

[6] TorchContributors, "Models and pre-trained weights," *PyTorch*, 2017-present. [Online]. Available: https://pytorch.org/vision/main/models.html

[7] ——, "Transforming and augmenting images," *PyTorch*, 2017-present. [Online]. Available: https://pytorch.org/vision/main/transforms.html

[1] [2] [3] [4] [5] [6] [7]