# Analysis of Vision Transformer’s Performance on Small Datasets

**JingYu Hu**
University of Toronto
jingyu.hu@mail.utoronto.ca

**Shizhuo Sun**
University of Toronto
shizhuo.sun@mail.utoronto.ca

**Wenxin Zhang**
University of Toronto
wenxin.zhang@mail.utoronto.ca

## Abstract

In this report, we will first implement and train a convolutional neural network model along with a vision transformer model on a small dataset. Then use saliency map to visualize two model’s performance, and suggest reasons of why vision transformer might perform better or worse than convolutional neural network. Finally we will re-implement a modified version of vision transformer and perform hyperparameter tuning.

## Introduction

With the article "Attention is All You Need"[8] being published in 2017, a new era of deep learning has started with varied models invented adopting attention mechanism[1, 9, 10]. In 2021 Google’s vision transformer was firstly introduced[1], and gained wide attention for its efficiency of training on extremely large image datasets with its high accuracy compared with traditional machine learning models.[1,4]

However as more machine learners start to learn and apply this model in daily tasks, they discovered the poor performance of vision transformer when applied on small datasets[3]. This problem draws our attention and we immediately started working on it, try to understand the reason why this issue occurs, and whether a possible improvement could be made.

## Related Work

### Transformer Architecture

Transformer architecture is mainly used to construct deep neural networks which compose of encoder-decoder layers with blocks where each one of them is implemented based on attention mechanism.

The key idea of attention is weighted output computation, which uses a well-chosen compatibility function and softmax activation to calculate each input’s relevance towards predicting the next output, and is represented as weight of attention output.[8] Thus higher-weight inputs will have significant impact on output’s generation, while inputs determined as "less impactful" by attention model contributes less towards new predictions.

Some common transformer methods involve ViT, TNT and Swin etc [11].

### Convolutional Neural Network

Preprint. Under review.

As opposed to neural network architectures with all layers being fully connected, convolution neural networks are mainly composed of convolution layer and pooling layer, where convolutional layer uses a small-sized kernel to apply on each part of input, by exploit locally sharing weight, and extract key features according to some criterion[12]. It has been widely adopted by further improvements; some examples include ResNet, AlexNet, GoogleNet etc[12].

### Saliency Map

Saliency map highlights the input image's components, for which provide most significant impact on how the model classifies the image into designated class. Its mechanism involves computing the one-class loss of a given image, perform backward pass on the given one-class loss to generate image gradient, and finally generate the grey-scale image where the intensity of each pixel is based on gradient value, which indicates the significance of current pixel for the model to classify the image as the given class. [7]

By visualizing the saliency map of a given image, it is possible to infer how a image is processed, and check whether the model could capture features that a human could easily detect.

## Methods

### Visualize Vit and CNN Using Saliency Map

In this part, we will first train a vision transformer model named "Vit_base_patch16_224"[3] with patch size 16 and input image size 224, and a convolutional neural network named "Wide_Resnet-50-2"[5]. The training will be conducted on a small dataset called "Intel Image Classification" dataset[6] containing approximately 24,000 images with size 150 from 6 classes. Then we will use saliency map[7] to visualize two trained models and hypothesize why convolutional neural network surpasses vision transformer on small datasets.

### Implement Modified Vit and Tune Hyperparameter

In this part, we will implement a modified vision transformer and perform hyperparameter tuning on training. The modification is introduced in the article "Vision transformer for Small-Size Datasets"[3], which introduced and applied "shifted-patch-tokenization" and "locally-self-attention mechanism"[3]. The result of the paper shows the final training accuracy of modified vision transformer model has improved validation accuracy by $3\%$ to $5\%$ [3].

Shifted-patch-tokenization's main idea is to shift the image on all four diagonal directions by half of patch size, then apply normalization and linear projection on the flattened concatenation on original image and all shifted image to acquire tokens, which is critical for embedding[4]. The algorithm box below provides structure of implementation. By doing this, the evaluation of each image patch will have overlaps with adjacent patches, thus could increase locally inductive bias of Vit [3,4].

```
Shifted Patch Tokenization(input_image, patch_size)
  for directions in up_left, up_right, down_left, down_right do
  shift input_image by direction with patch_size // 2
  fill the left space of shifted_image with colour 0
  end for

  concat_image ← concatenation of resulting images and input_image
  pad_size ← patch_size
  kernel ← patch_size
  extract patches from concat_image with pad_size and kernel
  flatten and normalize patches
  token ← linear projection of patches
  return token
```

Locally-self-attention is introduced to mask the diagonal entries for adjacency matrix when evaluating attention scores. Basically it is applied before calculating attention score and has the following form:

$$A_{i,j}^{mask}(x) = \begin{cases} A_{i,j}(x) & x \neq y \\ -\infty & x = y \end{cases}$$

$A_{i,j}$ in above formula is the value of adjacency matrix at position $(i, j)$[4].

By masking the diagonal entries of adjacency matrix, attention scores are forced to increase for different patches, which improves spacial association of patches when making predictions using the model [3].

## Experiments & Results

### Saliency Map Visualization of Vit and CNN

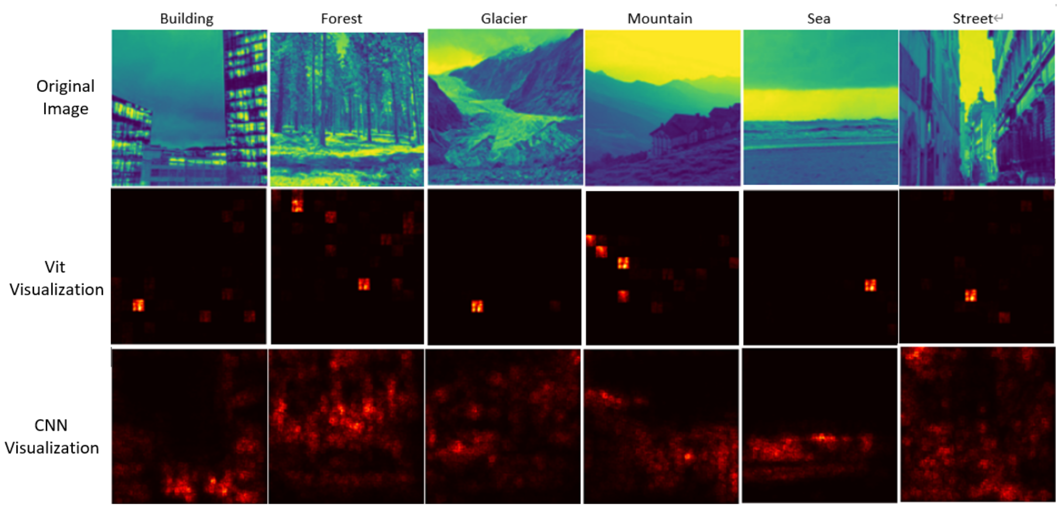


Figure 1: Saliency Map of Two Trained Models on Sample Images

We trained "vit_base_patch16_224" and "wide_Resnet-50-2" for 7 epoches with learning rate gradually decays from 1e-3 to 1e-5. Figure 1 shows the saliency map of two models on sample images.

As shown in visualization, CNN's activation is pixel-wise displayed, while Vit's high activation areas are shown as blocks of squares, where the contrast inside each square is relatively low. Those square blocks in Vit's saliency map actually represents patches' activation. Thus can conclude Vit evaluates images by treating each patch as a whole.

Above analysis suggests one weakness of vision transformer. As patch-wise evaluation can easily overlook details in an image due to having multiple class features in one patch, which could lower the activation of the patch for each class. Thus overall vision transformer cannot capture objects' shapes well enough to distinguish from other classes.

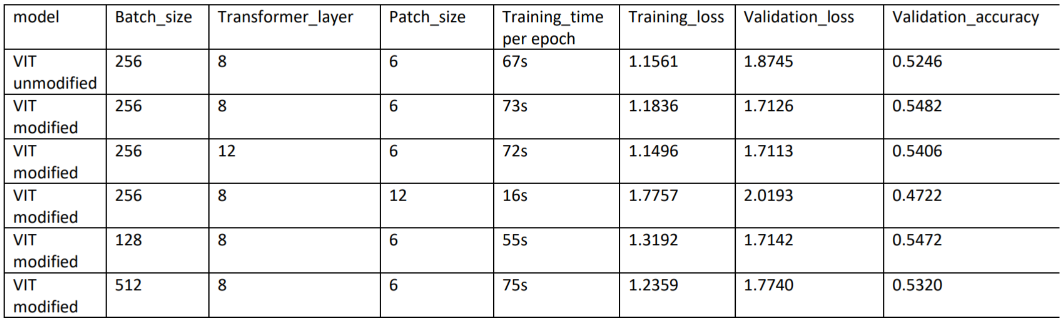
### Re-Implement Modified Vision Transformer and Perform Hyperparameter Tuning

We re-implemented the code which was inspired from the paper (reference). After observing the $2.5\%$ improvement on validation accuracy of modified vision transformer's performance, our group performed hyperparameter tuning on "number of transformer layers", "patch size" and "batch size". Figure 2 displays the resulting data of training for 50 epochs on each modified model, and appendix shows the loss and accuracy graph of during training for each model.

When the number of transformer layer of modified VIT model (abbreviate as MVIT) is increased from 8 to 12, a decrease of around $0.8\%$ in validation accuracy is observed, while the training loss is lower than unmodified VIT model and original MVIT. Possible reasons of validation accuracy drop could be due to model overfitting, which is shown in Figure 4e in "Appendix A" where model accuracy shows a trend of decreasing instead of flattening.

On the other hand, when patch size is increased to 12 for MVIT, the poor performance of tuned MVIT could be observed by both the final validation loss and accuracy from Figure 2, despite the significantly less training time for each epoch. This result is expected as increased patch size results in even worse grasp of object shape, due to the evaluation of Vit is patch-wise as analyzed in previous parts.

Finally we tuned batch size to 128 and 512. When batch size is 128, MVIT achieved significant progress in early stage of training which is indicated by the steep curve of first 10 epochs in Figure 4i. After that the loss curve starts to vibrate. However when batch size is 512, the accuracy curve grows slowly and does not have significant improvement on MVIT's accuracy. The comparison of time complexity for two models also indicates using large batch size for training is not efficient, as when batch size is 128, only 55 seconds is required for an epoch; while MVIT with batch size 512 needs at least 75 seconds.



| model | Batch_size | Transformer_layer | Patch_size | Training_time per epoch | Training_loss | Validation_loss | Validation_accuracy |
|---|---|---|---|---|---|---|---|
| VIT unmodified | 256 | 8 | 6 | 67s | 1.1561 | 1.8745 | 0.5246 |
| VIT modified | 256 | 8 | 6 | 73s | 1.1836 | 1.7126 | 0.5482 |
| VIT modified | 256 | 12 | 6 | 72s | 1.1496 | 1.7113 | 0.5406 |
| VIT modified | 256 | 8 | 12 | 16s | 1.7757 | 2.0193 | 0.4722 |
| VIT modified | 128 | 8 | 6 | 55s | 1.3192 | 1.7142 | 0.5472 |
| VIT modified | 512 | 8 | 6 | 75s | 1.2359 | 1.7740 | 0.5320 |

Figure 2: Table of Hyperparameter Tuning Result

## Discussion, Conclusion & Future Thoughts

Due to the limitation of computational resources, our experiment was unable to compare the performance of vision transformer and convolutional neural network on large datasets, which could easily lead to flawed conclusions. Besides, the modified version of vision transformer is not applied on Intel Image Classification dataset, which could be a possible weakness of our supporting evidence. Also, our visualization is based on saliency map only, but we haven't visualized the attention map as proposed in [1] to check whether the attention mechanism in Vit works properly.

In conclusion, while the Resnet (68 million) and Vit (86 million) have similar amount of parameters, CNN's performance on small-sized datasets is significantly better than vision transformer which typically requires training on extremely large datasets before tuning on middle-sized ones. Under a scenario lacking sufficient data and with only limited training resources, CNN would be more preferred.

However the extraordinary performance of vision transformer on large datasets indicates this model still plays an important role in research and development of advanced computer vision products. The training of this model is more preferred with a smaller patch size and batch size, to further enhance the model's performance.

On the other hand, based on the analysis that vision transformer evaluate images patch-wise, the possibility of developing non-linear patch for vision transformer which, instead of preserving shape, ensures only the area of each patch is the same. Exploration regarding similar topics has already started [13], and hopefully they could be applied to image recognition models, including vision transformer.

## References

1. Kolesnikov, A., Dosovitskiy, A., Weissenborn, D., Heigold, G., Uszkoreit, J., Beyer, L., ... Zhai, X. (2021). An image is worth 16x16 words: Transformers for image recognition at scale.
2. Masters, D., Luschi, C. (2018). Revisiting small batch training for deep neural networks. arXiv preprint arXiv:1804.07612.

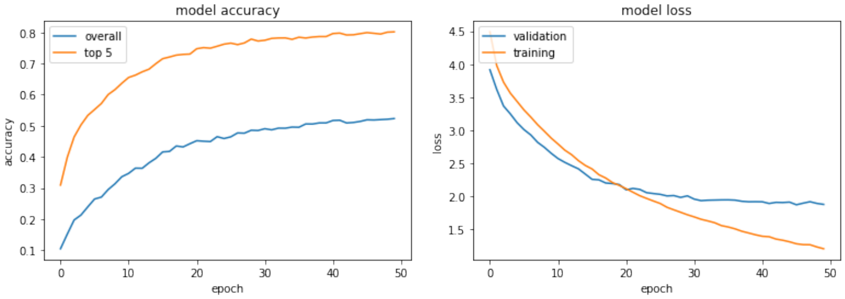
3. Lee, S. H., Lee, S., Song, B. C. (2021). Vision Transformer for Small-Size Datasets. arXiv preprint arXiv:2112.13492.
4. Gosthipaty, A. R. (2022, January 10). Keras Documentation: Train a vision transformer on small datasets. Keras. Retrieved April 20, 2022, from https://keras.io/examples/vision/vit_small_ds/final-notes
5. Zagoruyko, S., Komodakis, N. (2016). Wide residual networks. arXiv preprint arXiv:1605.07146.
6. Bansal, P. (2019, January 30). Intel Image Classification. Kaggle. Retrieved April 20, 2022, from https://www.kaggle.com/datasets/puneet6060/intel-image-classification
7. Simonyan, K., Vedaldi, A., Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034.
8. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.
9. Tetko, I. V., Karpov, P., Van Deursen, R., Godin, G. (2020). State-of-the-art augmented NLP transformer models for direct and single-step retrosynthesis. Nature communications, 11(1), 1-11.
10. Tetko, I. V., Karpov, P., Van Deursen, R., Godin, G. (2020). State-of-the-art augmented NLP transformer models for direct and single-step retrosynthesis. Nature communications, 11(1), 1-11.
11. Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., ... Tao, D. (2022). A survey on vision transformer. IEEE Transactions on Pattern Analysis and Machine Intelligence.
12. Aloysius, N., Geetha, M. (2017, April). A review on deep convolutional neural networks. In 2017 international conference on communication and signal processing (ICCSP) (pp. 0588-0592). IEEE.
13. Kim, H., Yoo, H., Lee, J. L., Lee, S. (2020). Convolution layer with nonlinear kernel of square of subtraction for dark-direction-free recognition of images. Mathematical Models in Engineering, 6(3), 147-159.

## Contributions

Each member in the group contributed to proposal, code implementation and final report equally.

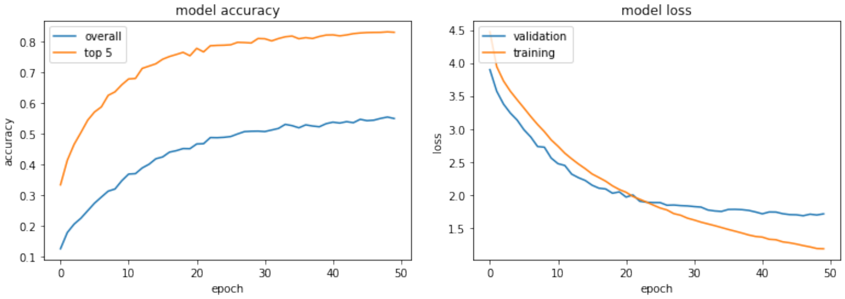
# Appendix A

Unmodified VIT



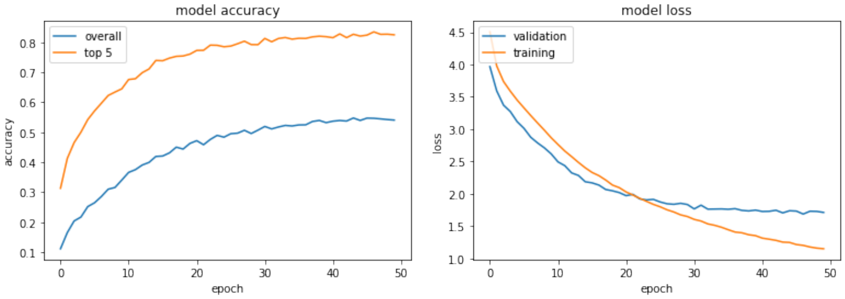
(a) (b)

Modified VIT with Original Hyperparameter Configuration

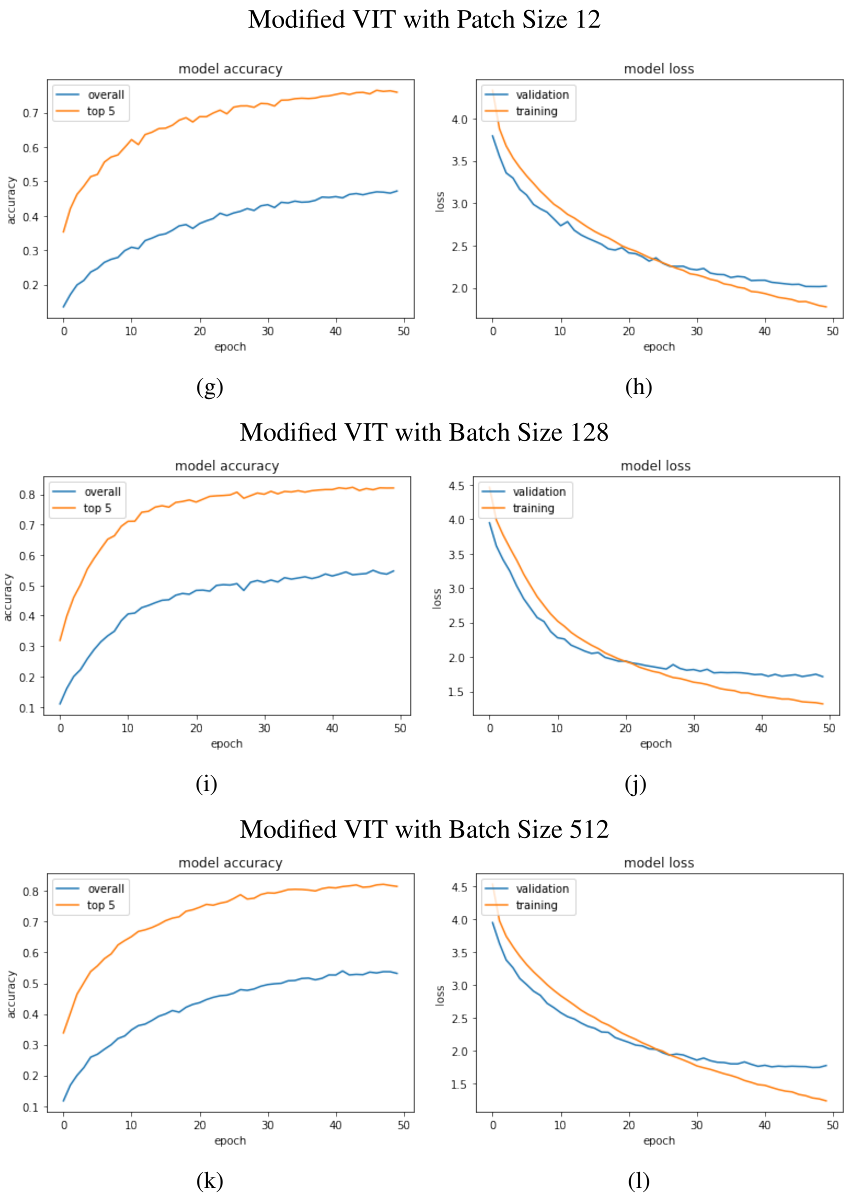


(c) (d)

Modified VIT with transformer layer 12



(e) (f)



Modified VIT with Patch Size 12

(g)

(h)

Modified VIT with Batch Size 128

(i)

(j)

Modified VIT with Batch Size 512

(k)

(l)

Figure 4: Loss and Accuracy of Phase 2's Training Results