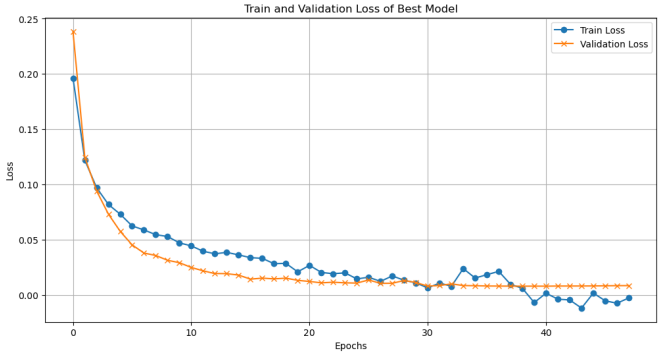
## Introduction

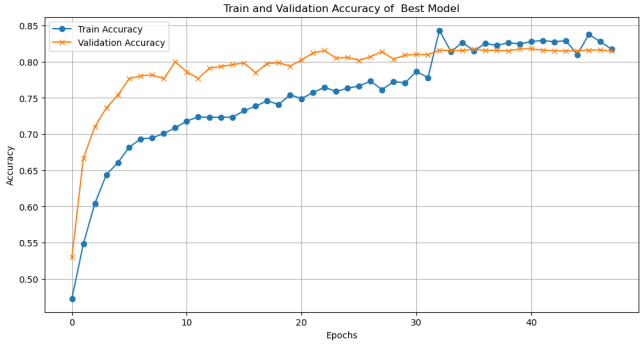
In this challenge, the goal is to identify the attribute labels of 1000 fashion photographs given 5000 training and 1000 validation samples. There are 6 categories, and each category contains more than one attribute label. There are a total of 26 labels.

## Specification

| | |
|---|---|
| Model | Swin (Hierarchical Vision Transformer using Shifted Windows) large Transformer |
| Loss Function | BCEWithLogitsLoss |
| Optimizer | Adam (lr=(1e-5, 5e-6, 1e-6), momentum=(0.9,0.999), wd=1e-5) |
| Number of Parameters | 197M |
| Machine | GPU RTX 4090(24GB) * 1<br>CPU12 vCPU Intel(R) Xeon(R) Platinum 8352V CPU @ 2.10GHz<br>Ram 90GB |

Results




| Best Epoch | | | Validation Loss | Validation Acc. | Best Test Score |
|---|---|---|---|---|---|
| 40 | 0.0016 | 0.82795 | 0.00374 | 0.81791 | **0.68191 (25)** |

## Dataset analysis and Preprocessing

### Augmentation

The Fashion Product Images Dataset contains only seven thousand samples which is relatively small compared to ImageNet -1K 1.3 million samples. Overfitting is unavoidable if directly fitting the original images to a complex neural network like Xception. Figure 1 shows

the validation accuracy of Xception model is stuck at 38.34% if data augmentation is not performed.


*Figure 2:Train and validation accuracy of Xception without*

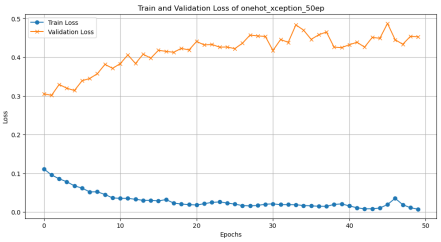



*Figure 1: Train and validation loss of Xception with augmentation*

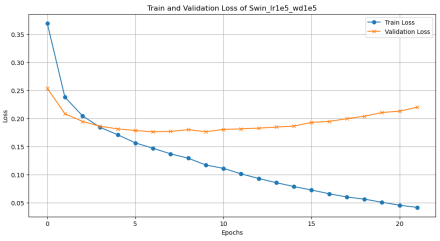


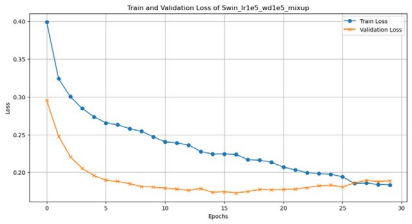

*Figure 3: Train and validation loss of Swin transformer with heavier augmentation*

The initial augmentation pipeline for cloth images includes **random horizontal flipping**, **random rotation** of +10 to -10 degrees, **color jittering** that adjusts the brightness, contrast, saturation, and hue, random cropping and resizing the image, and normalizing the pixel values to have a mean of 0.5 and a standard deviation of 0.5. **Normalization** would keep the value in the range 3 to -3, otherwise, large RGB values could lead to gradient explosion, causing instability during training. The mean and standard deviation are calculated from the image dataset with values [0.7657, 0.7359, 0.7254], and [0.2838, 0.2965, 0.3026].

After augmentation, the best validation accuracy of the Xception model spikes to 64.84%. However, according to Figure 2, the validation loss gained over the epoch rather than a declining trend like the training loss. Overfitting still exists, since it is a really small dataset. Thus, heavier augmentation is required.

The second stage of augmentation also added **random vertical flipping**, which is not too common due physical nature, but does not affect clothes. **Gaussian Blurring** is also added to simulate motion blur and reduce model reliance on fine detail. The validation loss in Figure 3 is closer to the training loss, despite still tilting upward.

Thirdly, **mixup** augmentation is added. It is simply done by adding two images with a random weight lambda. The labels are one-hot embeddings, so they are also added in the same way. The new loss now becomes the weighted sum of the loss between the predicted mix label and the original a and b label. Which can be formulated as $\lambda \cdot \text{loss}(\text{logit}, \text{label}_\text{a}) + (1 - \lambda) \cdot \text{loss}(\text{logit}, \text{label}_\text{b})$. Figure 4 reveals that mixup is a very effective augmentation technique, which the validation loss has the similar decreasing trend and is even lower than the training loss. Meaning that the overfitting problem is solved.




### Label Encoding

There are two methods to perform label encoding. Method A, directly using the class labels of each category, for instance [5,2,2,3,0,0]; or method B, ignoring the category and class hierarchy by using one-hot encoding for all classes, for example [0,0,0,0,0,1,0,0,1,0,0,1,0,0,0,0,1,1,0,0,0,0,0,1,0,0]. As such, different loss functions are required. A is more like a linear regression problem, so Mean Square Error loss would suit. B is a multilabel classification problem, so Binary Cross Entropy with Logits loss is needed. Cross entropy cannot be used due to the multilabel nature, the BCEWithLogitsLoss contains a in-build sigmoid layer. Considering the need to perform MixUp augmentation, one-hot encoding is the only choice.

More Augmentation,

Swin_large_heavieraug_wd

## Model Selection

Since the dataset contains only 6K images, it is insufficient to train a robust model from scratch. Therefore, pre-trained model is necessary for this task to make more efficient use of the limited data in a small dataset by leveraging the learned features from a larger dataset. In this task, only ImageNet-1K pre-trained models are used as required. It is well-known that CNN-based and ViT-based models are good at image classification, therefore, we wll experiment with Xception[1], InceptionV3 [2], Swin, and Swinv2 Transformer[3] for image classification. Xception stands out for its depthwise separable convolutions, optimizing parameter use. InceptionV3 excels with its deep network and inception modules, capturing features at various scales efficiently. Swin Transformer introduces hierarchical processing, dividing images into patches for better spatial information capture.

Without further ado, the three different architectures are tested with the default parameters, where the learning rate is set to $1 \times 10^{-4}$ for faster convergence. Figure 4, 5, 6 display the accuracy of Inceptionv3, Xception, and Swin transformer respectively. The Inceptionv3 has the lowest validation accuracy of 69.79%; Xception has 76.17%; and Swin transformer achieves the highest 77.59%. This could be attributed to Swin's global attention and hierarchical architecture that allows it to capture more complex pattern. Thus, Swin is chosen for this experiment.


*Figure 4: Train and validation accuracy of Inceptionv3*

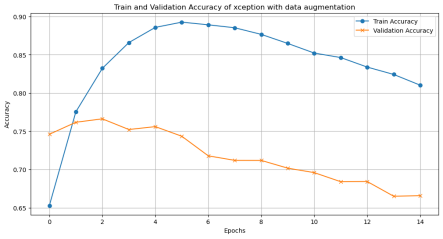



*Figure 5: Train and validation accuracy of Xception*

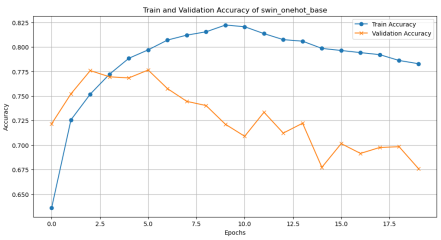



*Figure 4: Train and validation accuracy of Swin base transformer*

Besides, Swin transformer has different options for model sizes. We considered tiny, base, and large for this experiment. Each has parameter sizes of 29M, 88M, and 197M**.** We ran 20 epochs for each model with the above settings, and their test accuracies are 58.48%, 60.79%, and 62.66%. The result shows that a complex model that captures more features is beneficial. Large Swin transformer is chosen in this case.

## Optimization

In order to optimize the weights carefully, we have also experimented with different hyperparameters including learning rate, dropout rate, and weight decay.

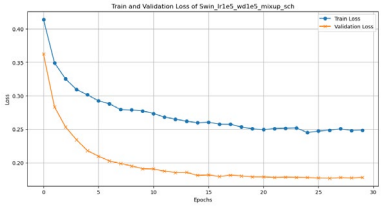
### Learning Rate Tuning

For learning rate, we tested with 1e-4, 1e-5 and 5e-6. The result is as in the table below.

| | 1e-4 | 1e-5 | 5e-6 |
|---|---|---|---|
| Validation acc. | 0.7777 | **0.7936** | 0.7900 |

### Learning Rate Scheduling

We also tested Cosine annealing.




### Regularization

| Dropout ratio | 0.35 | 0.55 |
|---|---|---|
| Validation acc. | **0.7777** | 0.7760 |

| L2 Regularization | 1e-5 | 2e-5 | 5e-4 |
|---|---|---|---|
| Validation acc. | 0.7936 | **0.8029** | 0.7900 |

Reference

[1] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” arXiv.org, https://arxiv.org/abs/1610.02357 (accessed Mar. 29, 2024).

https://arxiv.org/abs/2103.14030 (accessed Mar. 29, 2024).

[2] “Advanced guide to inception V3 | cloud TPU | google cloud,” Google, https://cloud.google.com/tpu/docs/inception-v3-advanced (accessed Mar. 29, 2024).

[3] Z. Liu et al., “Swin Transformer: Hierarchical vision transformer using shifted windows,” arXiv.org,