PA0 Assignment
Rabia Aslam
September 2025

Github Link: https://github.com/AimeeAyat/ATML/tree/PA0

Task 1: Inner Workings of ResNet-152

Introduction

The ResNet-152 model was developed to solve the problem of Deep Learning where deeper networks tend to suffer from vanishing gradient problems and thus reduce accuracy and increase loss when number of layers are increased. (Fig. 1) To solve the problem, concepts of Residual Connection and skip connection were introduced which made deep neural networks possible. Many newer convnets have replaced ResNet and it is now used as a reference only.

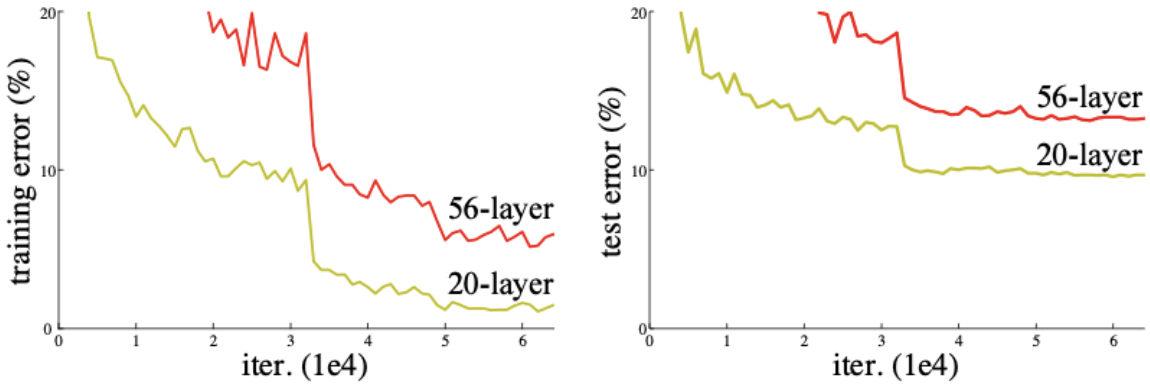


Fig (1.1)

Implications of training ResNet-152 from scratch

Resnet-152 is a very deep CNN with heavy compute. Training it from scratch on small datasets will cause models to memorize data. Therefore if one needs to use ResNet they should freeze all or most layers so that model can use its pretrained parameters which have already learnt about shapes and filters and different aspects of how to distinguish among images. Freezing layers let models use pretrained weights. It saves computations and overfitting.

Training and Validation Accuracy and Loss Comparison

Training loss decreases gradually from 0.7327 to 0.5695 and validation loss also decreases from 0.5853 to 0.4972 in first 7 epochs and then rises a little and stabilizes at 0.5654. Further training may have reduced it further. I was ablke to get 83.14% validation accuracy in just 10 epochs with all backbone freezed and only classification head active.

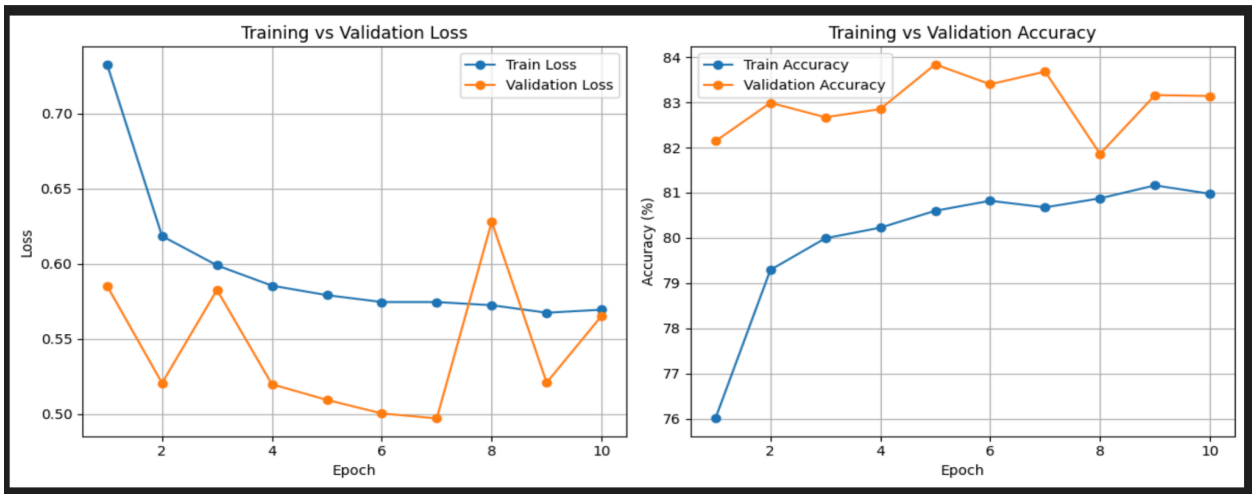


Fig (1.2)

Feature Hierarchies and Representations

I have evaluated the model architecture to understand the layers and how training different layers will impact the accuracy. t-SNE and UMAP representations of the dataset are shown in Fig (1.3 and 1.4). The architecture of layers with batch size 128 is:

Early ——–> (128, 256, 56, 56),
Middle ——-> (128, 1024, 14, 14)
Late ——-> (128, 2048, 1, 1)

Clear clusters are visible with full fine tuning which in turn has increased accuracy to 96.32%.

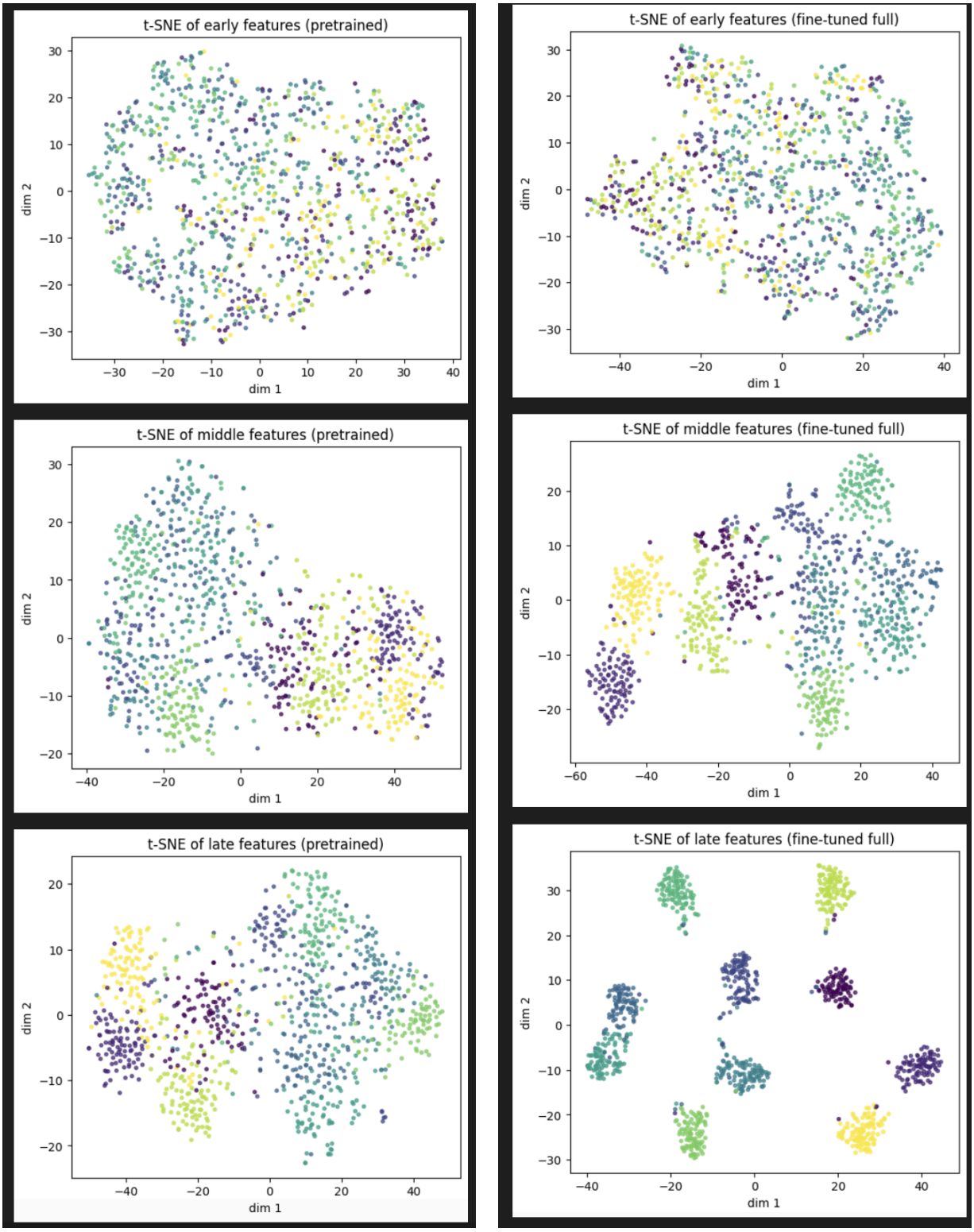

t-SNE of early features (pretrained)
t-SNE of early features (fine-tuned full)
t-SNE of middle features (pretrained)
t-SNE of middle features (fine-tuned full)
t-SNE of late features (pretrained)
t-SNE of late features (fine-tuned full)
dim 1
dim 2

Fig (1.3)

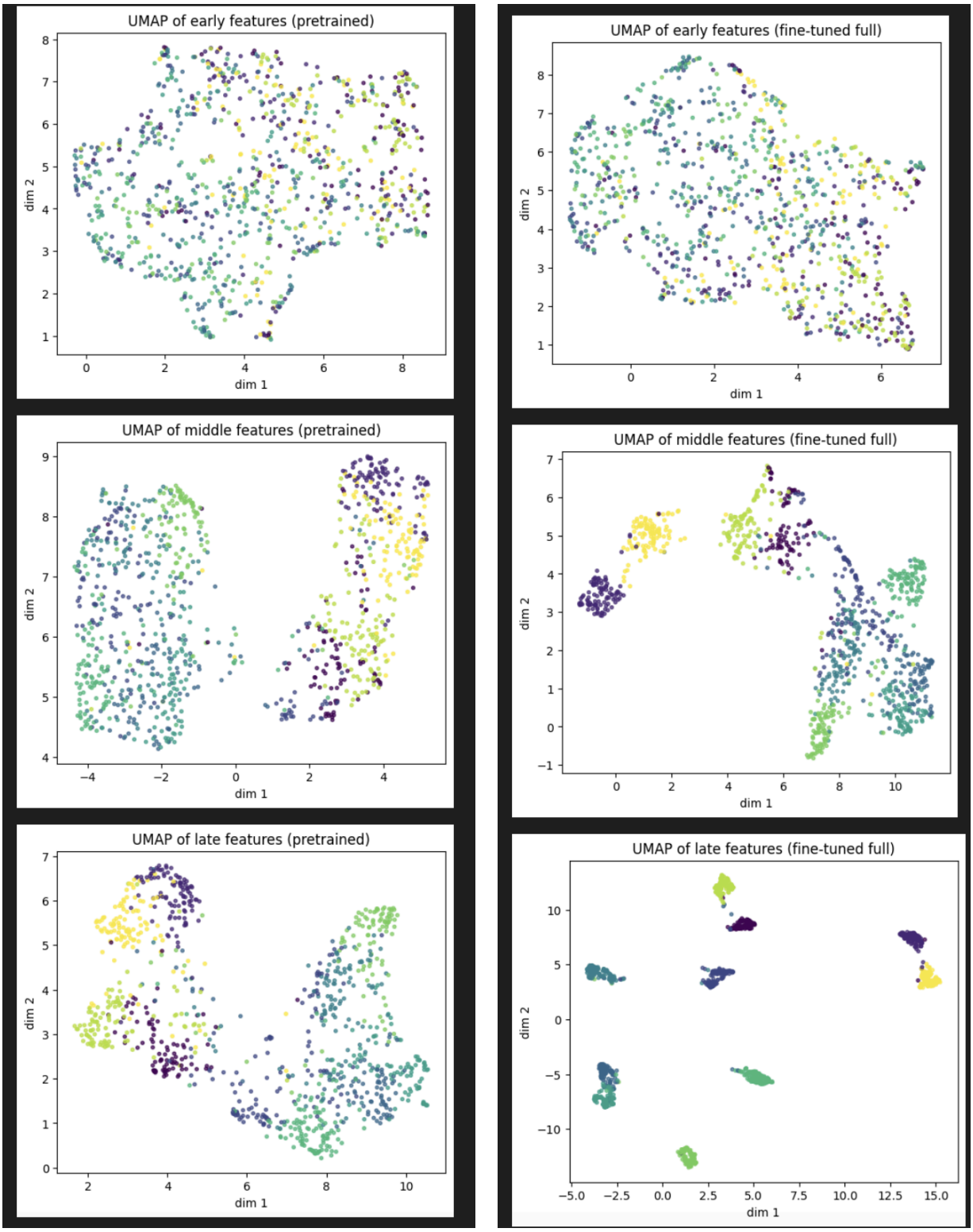

UMAP of early features (pretrained)
UMAP of early features (fine-tuned full)
UMAP of middle features (pretrained)
UMAP of middle features (fine-tuned full)
UMAP of late features (pretrained)
UMAP of late features (fine-tuned full)
dim 1
dim 2

Fig (1.4)

On the CIFAR-10 dataset, the results show a clear gap between using a pretrained model and starting from scratch. With linear probing, the pretrained model reached about 79.3% validation accuracy and 80.4% test accuracy, while random initialization performed poorly at only around 15% on both validation and test. Fine-tuning gave a big boost to the results, tuning just the last block improved accuracy to 91.0% on validation and 91.49% on the test set. Full fine-tuning worked best, achieving 96.34% validation accuracy and 96.32% on the test set. Overall, the results highlight how much pretrained models help and show that full fine-tuning leads to the strongest performance.

Task 2: Understanding Vision Transformers (ViT)

Introduction

Three small pretrained Vision Transformer (ViT) were used for image classification and attention visualization.

- google/vit-base-patch16-224
- facebook/deit-base-patch16-224
- google/vit-large-patch16-224

The models were pretrained on ImageNet. Images were resized to 224×224 before being fed into the model. The main goals were to

01. Record top-1 predictions and check if they make sense,
02. Visualize patch attention to see where the model looks,
03. Experiment with patch masking, and
04. Compare linear probes with different pooling methods.

Results using google/vit-base-patch16-224

Top-1 and Top-5 Predictions

We tested the model on multiple images, some results are:

1. A cat lying on the floor

Top-1: tabby, tabby cat (p=0.722)
Top-5:

1. tabby, tabby cat	p=0.722
2. tiger cat	p=0.164
3. Egyptian cat	p=0.067
4. Persian cat	p=0.030
5. lynx, catamount	p=0.007

True Label: tabby cat

Top-5 included other cat breeds and “Egyptian cat.” Shows the model was very confident it is a cat.

2. A chair in a room

Top-1: sliding door (p=0.087)
Top-5:

1. rocking chair, rocker	p=0.482
2. sliding door	p=0.087
3. studio couch, day bed	p=0.073
4. dining table, board	p=0.036
5. window shade	p=0.028

True Label: chair

Top-5 included rocking chair, studio couch,day bed, sliding door, dining table etc. This shows the model has attention both on door in the background and chair in foreground . Model is confusing background objects like (sliding door or window)

3. A red city bus/ Metro

Top-1: passenger car, coach, carriage (p=0.538)
Top-5:

passenger car, coach, carriage	p=0.538
orange	p=0.007
Granny Smith	p=0.002
banana	p=0.000
grocery store, grocery, food market, market	p=0.000

True Label: Metro Bus

Attention Visualization

Attention maps were created using the CLS token from the last layer (averaged across heads). These were upsampled to image size and overlaid as heatmaps.

Cat lying: Strong attention around the face
Chair: The model focused on seat edges, legs and room.
Bus: Attention concentrated on the front and road.
Cat with hat: The model concentrated attention on the hat and predicted the ferret.
Lemons: Multiple spots lit up, especially shiny areas.

This shows that ViT's have divided attentions across images like the cat with hat was wrongly predicted as ferret but had cat in top5. So ViTs generally pays attention to semantically meaningful parts, even when unusual elements (like the cat's hat) are present. With better training or models the accuracy can be increased.

A List of Results of all three models

== google/vit-base-patch16-224 ==
Sample 0: Top-1 -> **basset**, basset hound (0.26) | CIFAR10 label: **dog**
Sample 1: Top-1 -> tailed **frog**, bell toad, ribbed toad, tailed toad, Ascaphus trui (0.98) | CIFAR10 label: **frog**
Sample 2: Top-1 -> warplane, military **plane** (0.06) | CIFAR10 label: **airplane**

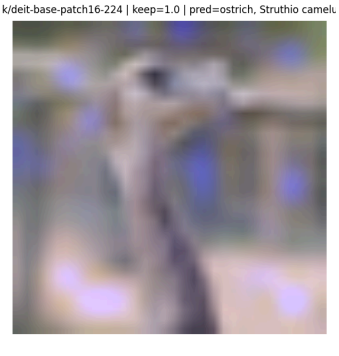
Sample 3: Top-1 -> f**ox squirrel**, eastern fox squirrel, Sciurus niger (0.17) | CIFAR10 label: **bird**
Sample 4: Top-1 -> moving **van** (0.81) | CIFAR10 label: **truck**
Sample 5: Top-1 -> **ostrich**, Struthio camelus (0.98) | CIFAR10 label: **bird**

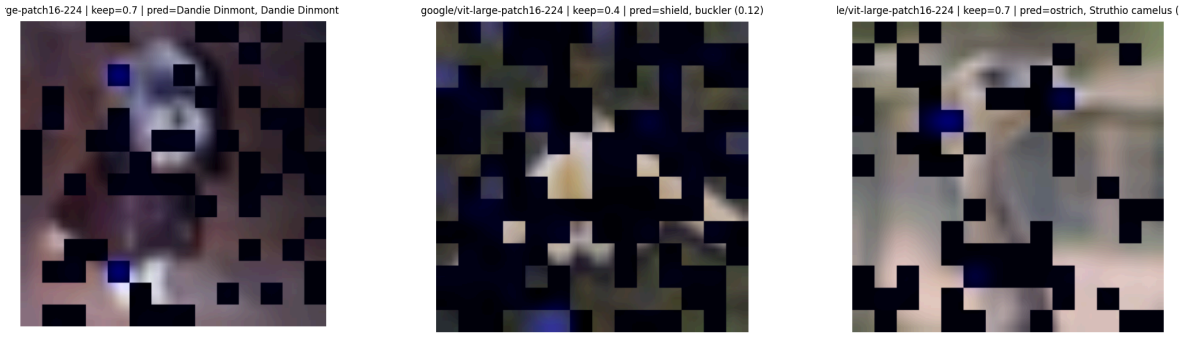
== facebook/deit-base-patch16-224 ==
Sample 0: Top-1 -> **Shih-Tzu** (0.16) | CIFAR10 label: **dog**
Sample 1: Top-1 -> tailed **frog**, bell toad, ribbed toad, tailed toad, Ascaphus trui (0.24) | CIFAR10 label: **frog**
Sample 2: Top-1 -> **panpipe**, pandean pipe, syrinx (0.04) | CIFAR10 label: **airplane**
Sample 3: Top-1 -> **brambling**, Fringilla montifringilla (0.13) | CIFAR10 label: **bird**
Sample 4: Top-1 -> moving **van** (0.08) | CIFAR10 label: **truck**
Sample 5: Top-1 -> **ostrich**, Struthio camelus (0.14) | CIFAR10 label: **bird**

== google/vit-large-patch16-224 ==
Sample 0: Top-1 -> **Dandie Dinmont**, Dandie Dinmont terrier (0.64) | CIFAR10 label: **dog**
Sample 1: Top-1 -> tailed **frog**, bell toad, ribbed toad, tailed toad, Ascaphus trui (0.99) | CIFAR10 label: **frog**
Sample 2: Top-1 -> **corkscrew**, bottle screw (0.34) | CIFAR10 label: **airplane**
Sample 3: Top-1 -> **jacamar** (0.21) | CIFAR10 label: **bird**
Sample 4: Top-1 -> moving **van** (0.71) | CIFAR10 label: **truck**
Sample 5: Top-1 -> **ostrich**, Struthio camelus (0.97) | CIFAR10 label: **bird**

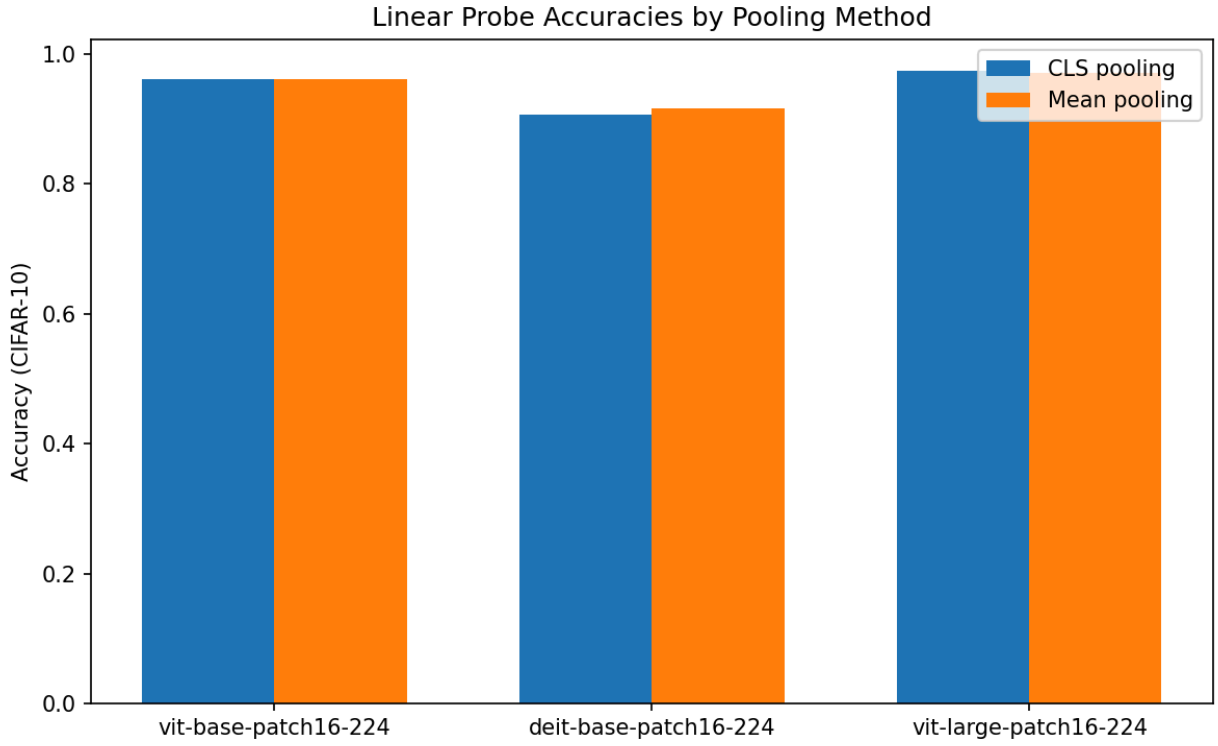
Experimenting with Masking Patches

We tested how robust ViT is to missing patches. We experimented with 70% and 40% of image revealed and the rest masked as shown.





Accuracies were evaluated for all three models on CIFAR-10 dataset using a linear probe (features frozen) .CLS pooling and Mean pooling results were compared



The bar chart shows the probe accuracies. Model size is most important as ViT-large has 97.4 cls and 97.2 mean pooling accuracy, while ViT-Base has same accuracy for cls and mean pooling i.e. 96.2 further DeiT base has performed the least with cls 91.75 and mean pooling accuracy to be 90.5 so bigger models give features that are easier for a linear classifier to separate. From the results CLS performs better for google ViTs.

Task 4: Training Variational Autoencoders

Variational Auto Encoders learns probabilistic gaussian latent distribution of data. The encoder maps images to the latent space. The decoder reconstructs images from the latent distribution. Learning is made possible through the use of the reparameterization trick. I have explored different techniques to reconstruct the best representation of images. This includes exploring active z-dimensions, role of KL and reconstruction loss on regenerations and sampling from different distributions i.e. normal or laplacian.

Training VAE

I have trained a VAE on Fashion-MNIST with a Gaussian decoder (MSE) and a standard Normal prior. Loss curves show stable optimization: reconstruction error fell from 28.63 to 14.88 MSE per sample while the KL term rose from 6.13 and plateaued at approx. 7.98 nats, yielding a total ELBO around 22.86 as in Fig. (4.1)

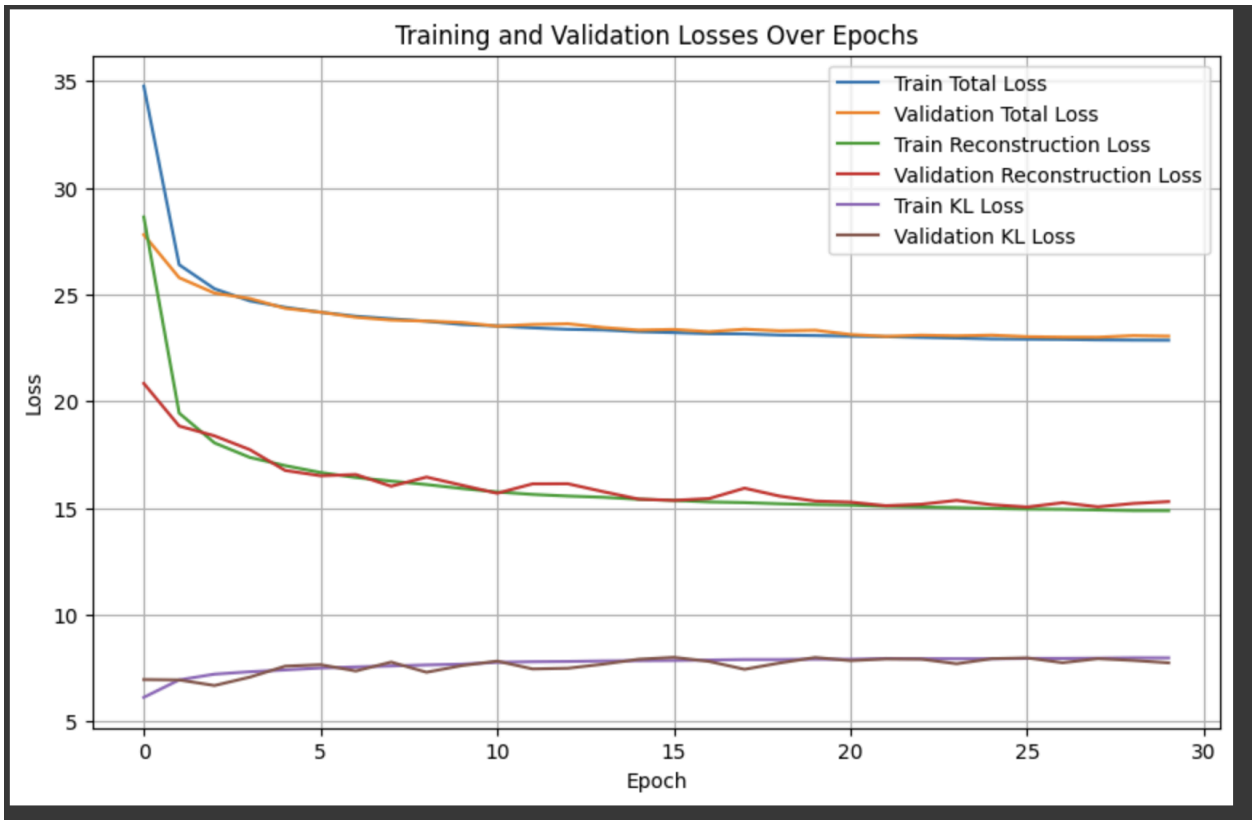


Fig (4.1)

Visualize Reconstructions and Generations

Reconstructions are compared with originals and it was observed that reconstructions are blurry and only preserves edges and silhouettes. We can see that textures and texts are all gone as Gaussian approximates pixels.

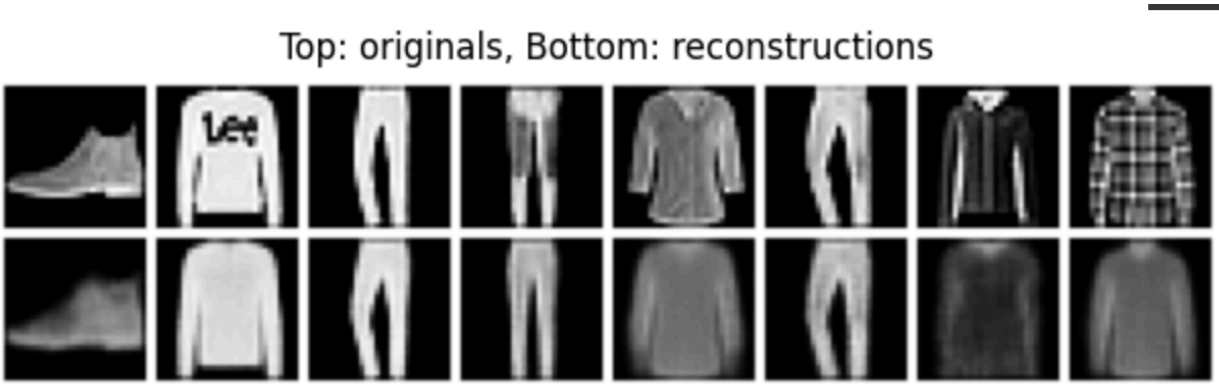


Fig (4.2)

3. Gaussian vs laplacian generations

Sampling from the Normal prior has produced low quality images, the reason could be a small number of epochs and because most dimensions were unused and only 6 out of 20 dimensions have contributed to the results. The encoder did not have enough to learn, sampling from a Laplace prior also degrades quality due to prior-decoder mismatch. Since images are reconstructed from Gaussian latent space of 20dim, the images are blurry and edges aren't refined.

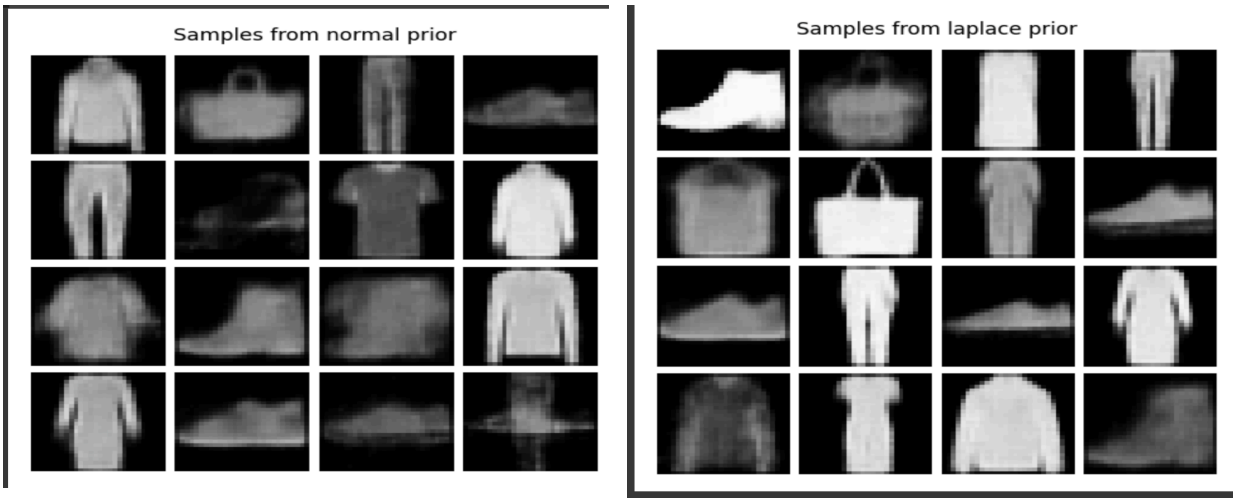
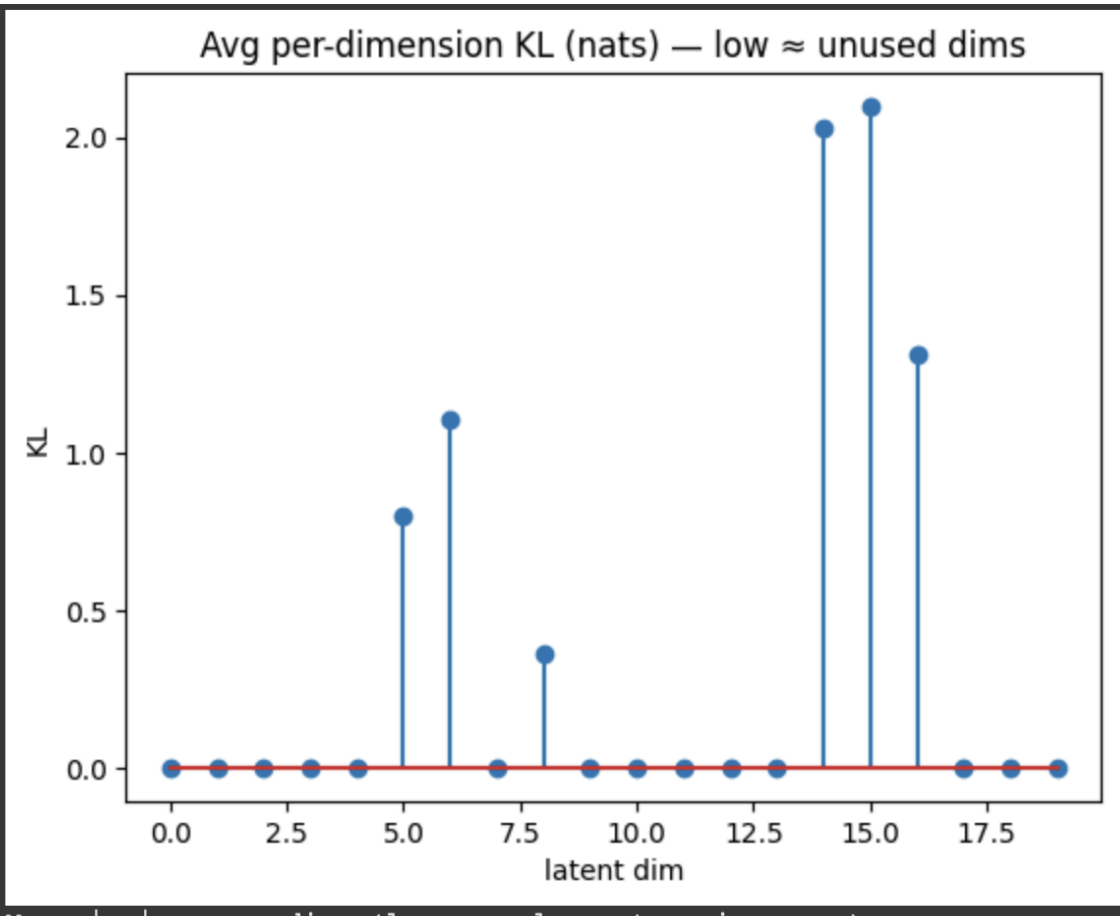


Fig (4.3)

4. Posterior Collapse

Plot of Fig (4.2) clearly shows which latent coordinates are used by model. For most dimensions KL is zero and only 6 dimensions show a spike with considerable KL values. The stats show that mean per dimension is small and log per variance is highly negative for the active dimensions. From the stats we can see that KL is largely variance-driven.




```
Mean |mu| across dims (lower ≈ closer to prior mean):
[0.01753526 0.04362604 0.0040834  0.0031585  0.02009211 0.07844752
 0.03521003 0.02037184 0.06009704 0.02184301 0.00939641 0.02512023
 0.01715139 0.03444309 0.07379159 0.04893012 0.0531049  0.01131726
 0.00419823 0.02459726]
Mean logvar across dims (≈0 means var ~ 1):
[-2.0975057e-02 -1.3847530e-02 -5.6832889e-03 -7.7015832e-03
 -2.6408961e-02 -1.6777885e+00 -2.2694016e+00 -2.0041941e-03
 -7.3168087e-01  2.5753400e-03 -2.2047838e-02 -2.9426817e-02
 -8.5356021e-03 -2.3386558e-03 -4.1056337e+00 -4.1835051e+00
 -2.6950746e+00 -7.3109358e-04 -4.6359818e-02 -6.1994134e-03]
Total KL per sample (approx): 7.724 nats
```

Fig(4.4)

4. Mitigating Posterior Collapse

To mitigate the effect of posterior collapse techniques of warm-up = 10 and free bits = 0.05 are introduced. This was to give more room for the encoder to learn and map images in latent space. Resultantly images reconstructed have better quality and instead of 4 dimensions now all 20 dimensions are used.

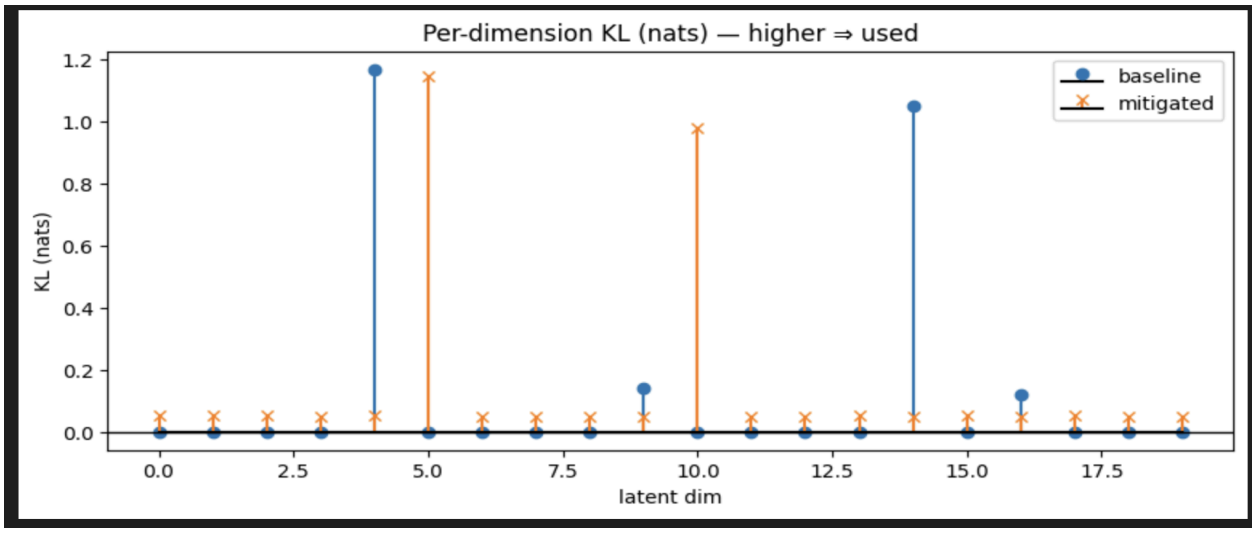


Fig (4.5)

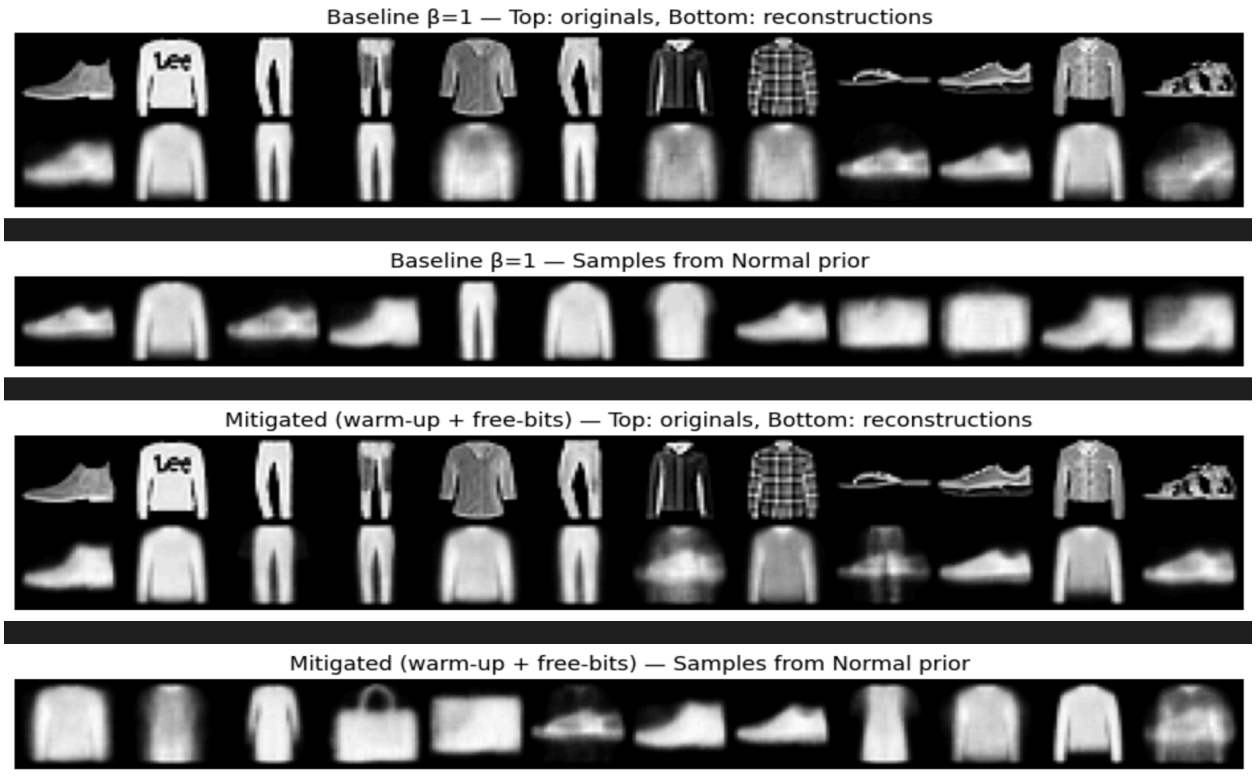


Fig (4.6)

Conclusion:

Per-dimension analysis reveals partial posterior collapse, despite a 20-D latent, only 5–6 dimensions carry substantial KL, largely via variance shrinkage (means near zero, strongly negative log-variances on active dims), therefore reconstructions preserve category and structure but lose fine details.WarmUp and free bits were introduced to enhance the quality of reconstructed images. Reconstruction is consistent with the information budget and the Gaussian objective. Overall, the model learns a compact, smooth latent representation that supports plausible generations and reconstructions.

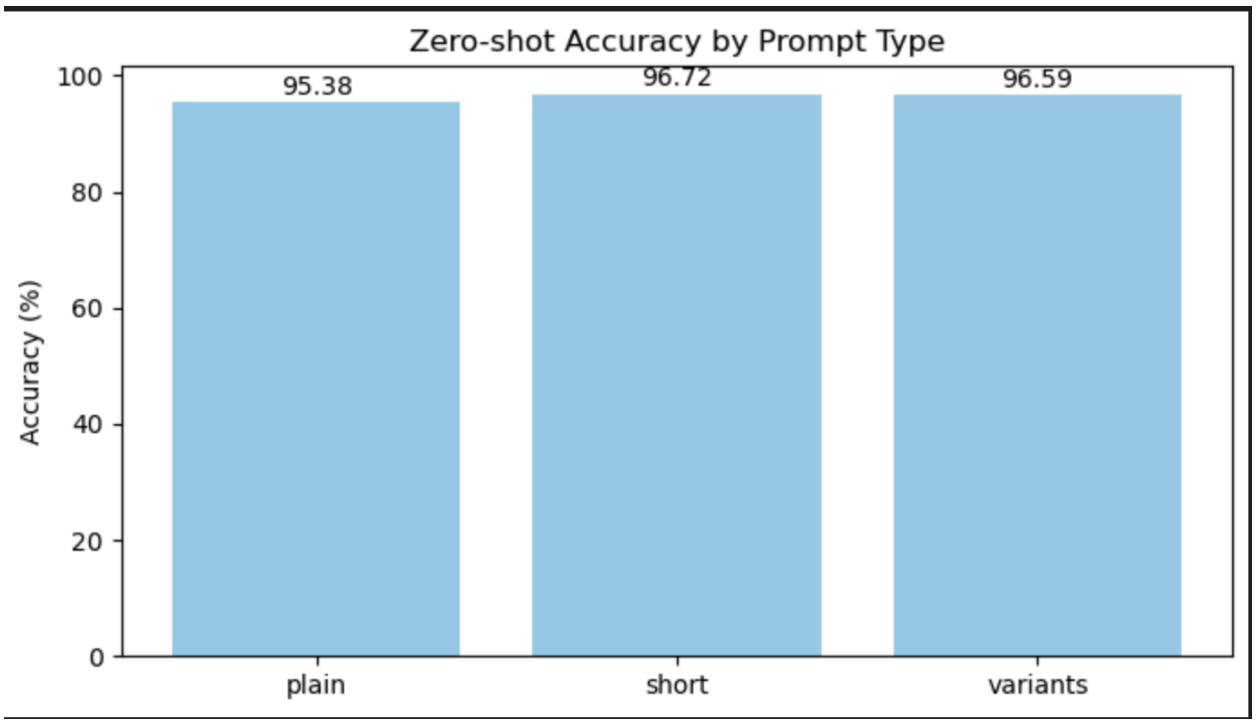
Some other possible ways to improve the model can be:

1) Change of objective function, we can try using BCEwithlogits on FashionMNIST dataset as Bernoulli can preserve edges and details better than MSE
2) Increasing dimensions of latent space. It may increase KL but can reduce reconstruction error

Task 5: Modality Gap in CLIP

1. Zero-Shot Classification on STL-10

Zero-shot accuracy using different prompting techniques was calculated. The results are shown in Fig (5.1)



(Fig 5.1)

Accuracies follow the trend: Short > Variants > Plain. Since CLIP was trained on large-scale web-scraped captions, which often appear in short or variant style, performance is better for these than plain raw labels.

Exploring the Modality Gap

CLIP is a multimodal model with separate encoders for text (Transformer) and images (Vision encoder). Due to differences in representation spaces and inherent noise in web data (e.g., incomplete or misleading captions), a modality gap arises. Ideally, embeddings from text and images should live in the same semantic space so that “a photo of a dog” and a dog image are close together. In practice, they occupy different regions. This problem is addressed using dimensionality reduction (t-SNE, UMAP) and alignment techniques (Procrustes with and without L2 normalization).

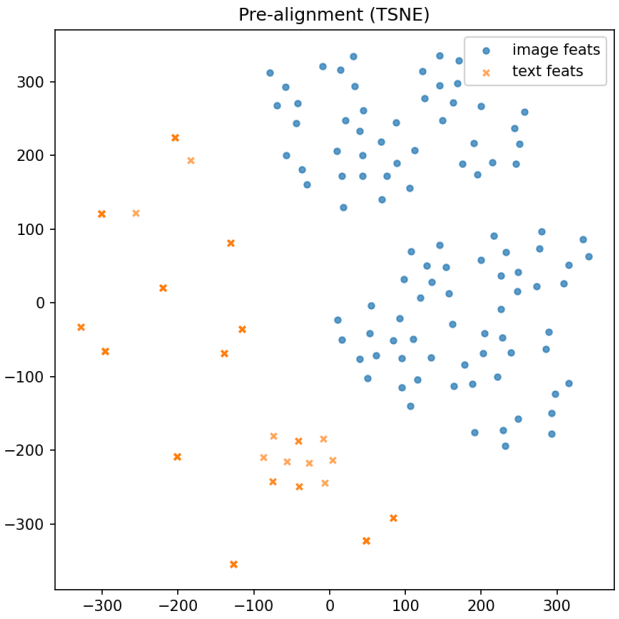


Fig (5.2)

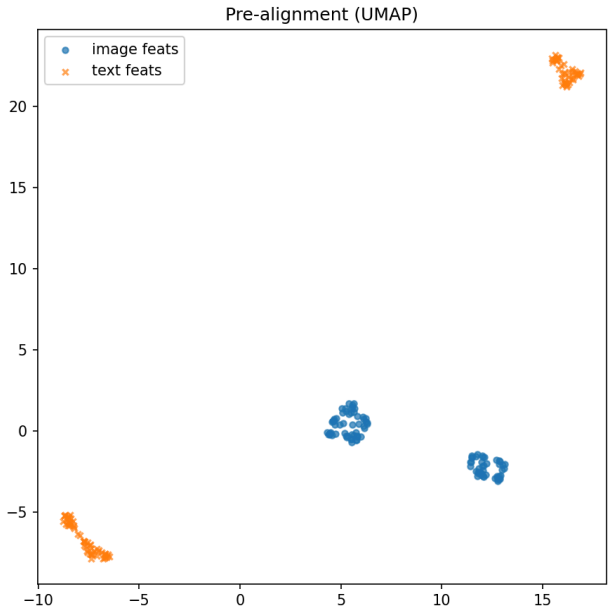


Fig (5.3)

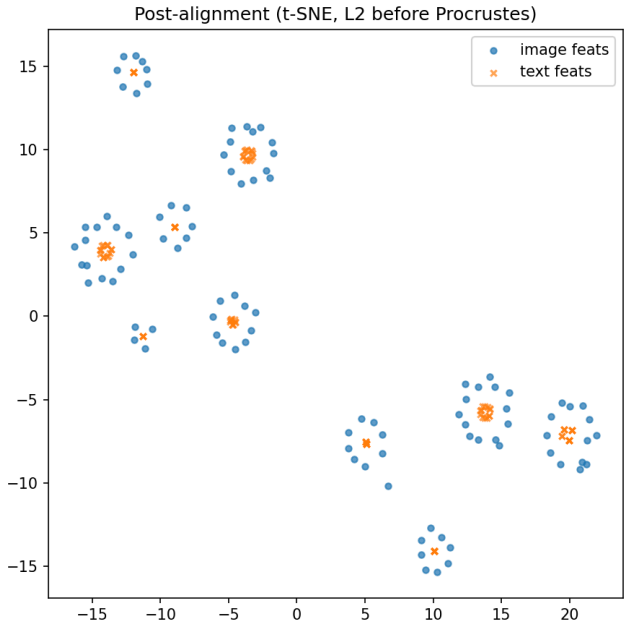


Fig (5.4)

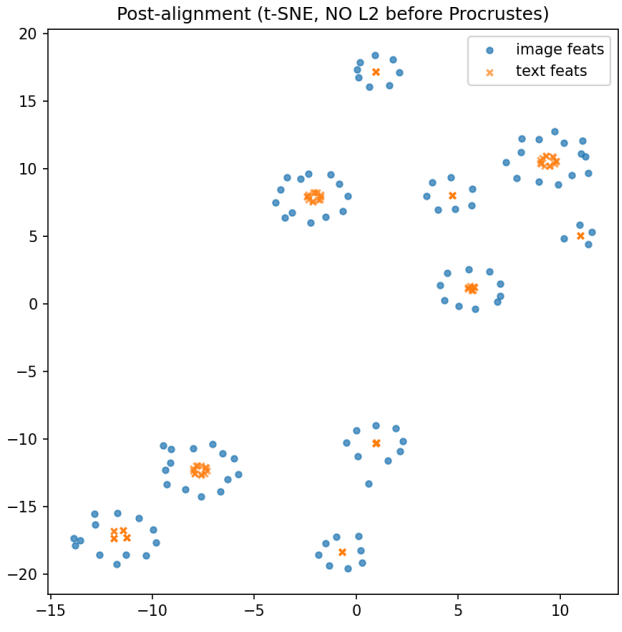


Fig (5.5)

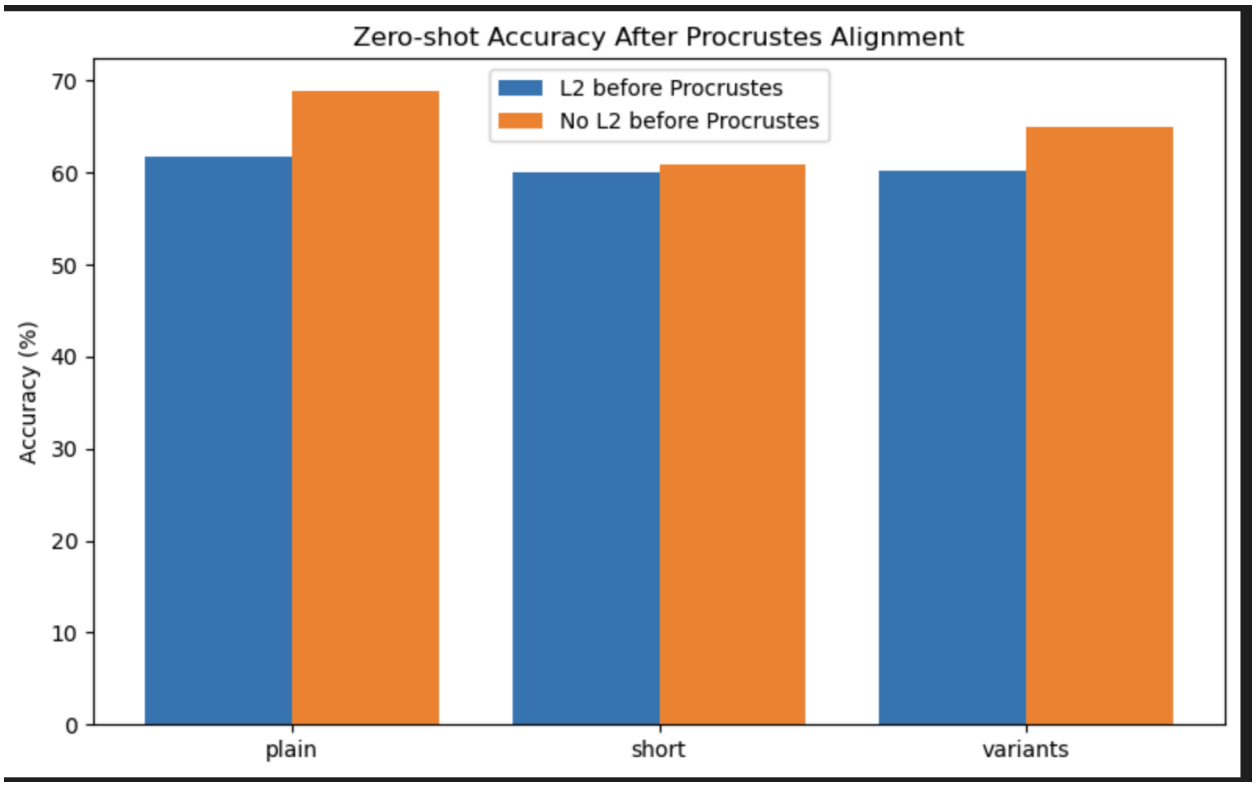
Findings

Pre-alignment: Images and texts are clearly separable forming distinct clusters in Pre-TSNE (Fig 5.2) and Pre-Umap alignment (Fig 5.3)
Post-alignment (Procrustes, with and without L2 normalization): Clusters overlapped significantly, indicating reduced modality gap.

Cosine Similarity

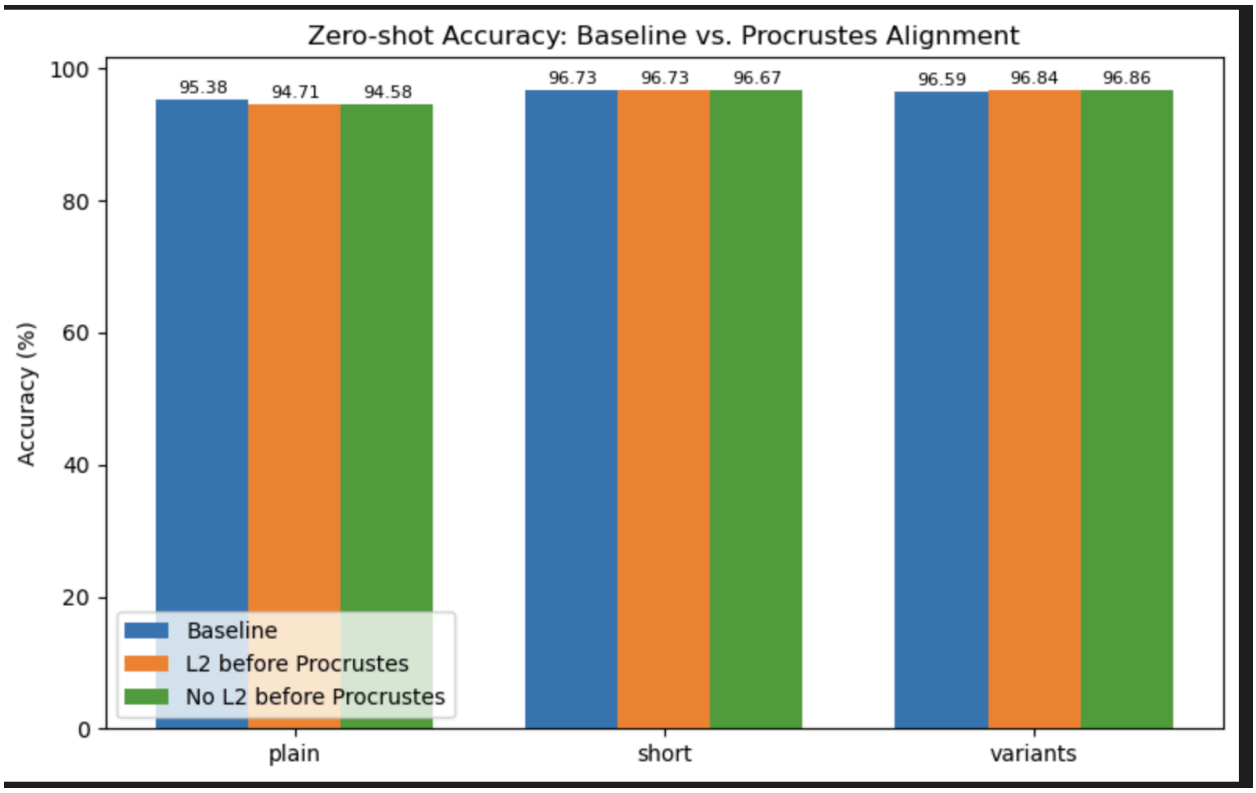
Mean cosine similarity was observed which came out to be:
Mean-Cosine-Pre: 0.2704313099384308
Mean-Cosine-Post-L2: 0.8707431554794312
Mean-Cosine-Post-NoL2: 0.8707432746887207

Alignment increases similarity between paired embeddings and improves cluster overlap, but reduces accuracy when only images are aligned (Fig. 5.6). When both images and prompts are aligned, accuracy remains high (Fig. 5.7).



(Fig 5.6)

If alignment is applied consistently to both images and prompts then accuracy stays high as shown in Fig (5.7)



(Fig 5.7)

Conclusion

CLIP performs well even with modality gap, due to preserving semantic structures during pretraining. Alignment further reduces modality gap but needs consistent application across both modalities to avoid hurting zero-shot accuracy.