Mini Project 2
Report

Praveen Patel
241110404

Uddeshya Raj
241110406

Puspesh Kumar Srivastava
241110405

Bhavya Garg
220296

1 Tasks 1.1 and 1.2

Objective: To find a suitable feature extraction technique that has high accuracy on datasets and develop an updation technique to perform incremental learning and overcome catastrophic forgetting.

1.1 Feature Extraction

We have used the google/vit-base-patch16-224 model(ViT) pre-trained on ImageNet-21k (a collection of 14 million images and 21k classes) for feature extraction. It achieves 99.5% accuracy when fine-tuned on CIFAR-10. The Vision Transformer (ViT) architecture divides images into fixed-size patches (e.g., 16x16 pixels), which are then flattened and linearly embedded into vectors. These embeddings are combined with positional encodings to retain spatial information. They are input into a standard Transformer encoder comprising multiple layers of multi-head self-attention and feed-forward neural networks. A special classification token ([CLS]) is appended to the sequence, and its representation after encoding is used for classification tasks.

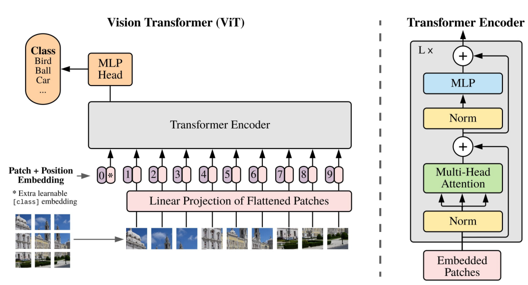

Figure 1: ViT Architecture

1.2 Updation Technique

Main Methodology:

Generative Sampling: We progressively stored the mean and covariance of the feature vectors extracted from image samples of each given/predicted class and created a class-wise normal distribution for each dataset. 250 synthetic points per class per dataset were generated from those normal distributions and added to the current dataset being processed. This new dataset which contains both the predicted labels and generated synthetic labels is then used to update the model to the next iteration and generate the mean and covariance to generate the normal distribution for the next iteration. The addition of synthetic data points addressed the issue of catastrophic forgetting and the accuracy of earlier datasets was almost the same throughout and did not degrade significantly.

Other Methodologies that we tried:

1. **Confidence-based update**: Conf_Val = 1/(1+d(x, mean, cov)), where d is Euclidean, Wasserstein, or Mahalanobis distance. In this method, only data points with confidence values above a carefully set threshold were used to update the mean and covariance of the next model.
2. **Regularization-based update**: Here, k is the regularization weight, N_i is the current sample count, n_i is the number of new samples, p_i is the current prototype, and $\mathbf{x}_j$ are the new sample vectors.

$$\mathbf{p}_i^{\text{new}} = (1-k)\mathbf{p}_i + k \cdot \frac{N_i \mathbf{p}_i + \sum_{j=1}^{n_i} \mathbf{x}_j}{N_i + n_i}$$

3. **No Learning**: We also tried storing mean and covariances at each level and using it to classify the subsequent dataset using all of the previous values separately. This methodology gave the most consistent results. Surprisingly, the best results for D1_eval to D10_eval were obtained when we used just euclidean distance between mean of D1 and data-points of every other datasets D2 to D20. That is, when our model didn't learn anything new from the upcoming datasets. In all other methods there was decline in accuracy for every subsequent models and datasets.
4. **De-Noise Technique**: The datasets D11 to D20 appear to be same with differences just in saturation, sharpness and noise. So we tried to denoise the images of dataset D11 and D12 and then generate their feature vectors. The classification accuracy resulted around 10% for both. We had used Wiener filter to denoise a noisy image.

Note: We have attached a single ipynb file that works for both Task 1.1 and Task 1.2, as we have used same methodology for both Tasks. Also we have attached a separate ipynb used for feature extraction, and code files of other accompanying techniques have also been added.

Link to Extracted Features along with ipynb files Dropbox

Link to Video Presentation: Youtube In the video, we mistakenly referred to our group number as 91, but it is actually **93**. We apologize for the error.

1.2.1 Results of f1 to f10

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
f_1	95.84	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
f_2	95.96	95.64	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
f_3	96.00	95.76	95.36	0.00	0.00	0.00	0.00	0.00	0.00	0.0
f_4	95.88	95.68	95.36	95.68	0.00	0.00	0.00	0.00	0.00	0.0
f_5	95.84	95.68	95.36	95.68	95.72	0.00	0.00	0.00	0.00	0.0
f_6	95.84	95.64	95.36	95.76	95.72	95.96	0.00	0.00	0.00	0.0
f_7	95.76	95.64	95.40	95.84	95.76	96.00	95.40	0.00	0.00	0.0
f_8	95.76	95.64	95.40	95.84	95.76	96.00	95.36	95.28	0.00	0.0
f_9	95.80	95.68	95.44	95.84	95.80	96.00	95.36	95.32	96.16	0.0
f_10	95.80	95.68	95.44	95.84	95.80	96.00	95.36	95.36	96.16	96.0

Figure 2: Similar Distribution Dataset

1.2.2 Results of f11 to f20

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20
f_11	95.84	95.64	95.28	95.80	95.64	95.96	95.36	95.48	96.04	95.96	82.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
f_12	95.72	95.60	95.24	95.76	95.64	95.84	95.36	95.48	96.08	95.92	82.04	73.56	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
f_13	95.72	95.60	95.24	95.76	95.64	95.84	95.36	95.40	96.04	95.96	82.04	73.60	88.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00
f_14	95.72	95.60	95.24	95.76	95.64	95.84	95.36	95.40	96.04	95.96	82.04	73.60	88.08	92.52	0.00	0.00	0.00	0.00	0.00	0.00
f_15	95.72	95.60	95.24	95.76	95.64	95.84	95.36	95.40	96.04	95.96	82.04	73.60	88.08	92.52	94.40	0.00	0.00	0.00	0.00	0.00
f_16	95.72	95.60	95.28	95.76	95.68	95.88	95.36	95.40	96.00	95.96	82.04	73.60	88.08	92.52	94.52	87.84	0.00	0.00	0.00	0.00
f_17	95.72	95.60	95.28	95.64	95.64	95.88	95.36	95.44	95.96	95.96	82.00	73.60	88.08	92.56	94.48	87.84	83.84	0.00	0.00	0.00
f_18	95.76	95.64	95.28	95.64	95.64	95.88	95.36	95.44	95.92	95.96	82.00	73.60	88.04	92.56	94.44	87.84	83.84	84.44	0.00	0.00
f_19	95.76	95.64	95.20	95.64	95.64	95.88	95.36	95.40	95.88	95.96	82.00	73.60	88.04	92.56	94.44	87.92	83.72	84.44	78.04	0.00
f_20	95.76	95.64	95.20	95.64	95.64	95.88	95.36	95.40	95.88	95.96	81.96	73.60	88.04	92.56	94.44	87.88	83.72	84.44	78.04	92.32

Figure 3: Different Distribution Dataset

References

1 Rectification-Based Knowledge Retention for Task Incremental Learning

2Incremental Prototype Tuning for Class Incremental Learning

3Prototype Augmentation and Self-Supervision for Incremental Learning

4ViT-base-patch16-224

5PILoRA: Prototype Guided Incremental LoRA for Federated Class-Incremental Learning