Classification ML Project

Aaditi Agrawal
220006

Vasudharaje Srivastava
221176

Manvi Kumar
220630

Suneet Maharana
221105

Ritesh Baviskar
220286

Aditya Jagdale
220470

1 Task-1.1

The datasets consisted of images from the CIFAR dataset. The first dataset D_1 is labelled while all the others are unlabelled. We perform supervised machine learning on D_1 and unsupervised on the rest of the datasets.

Model Overview

The model in this paper consists of three key components: a Vision Transformer (ViT) based feature extractor, a Bayesian classifier, and a training procedure that incorporates pseudo-labeling for unlabelled data.

1.1 Vision Transformer (ViT) Feature Extraction

The feature extraction is carried out using a pretrained Vision Transformer (ViT) model from Hugging Face, specifically the model `google/vit-base-patch16-224-in21k`. ViT divides an input image into fixed-size patches, flattens them, and passes them through a series of transformer encoder layers. The key feature of ViT is its ability to capture long-range dependencies across patches in the image through the self-attention mechanism.

The input image is first resized to 224×224 and normalized. After the image passes through the transformer layers, the output consists of a sequence of embeddings corresponding to each patch. The embedding for the special `[CLS]` token is extracted, which serves as a compact representation of the entire image. This embedding is then passed as input to the Bayesian classifier for further processing.

1.2 Bayesian Classifier

The Bayesian classifier uses a probabilistic framework where the posterior distribution of each class is computed given the features extracted from the images. For each class c, the classifier maintains a prior mean μ_c and a prior covariance Σ_c, which are updated through Bayesian inference as new data becomes available.

The update for the posterior distribution of each class is computed using the Bayesian update rule. Given the current class features $\mathbf{X}_c$ and the prior parameters, the posterior mean μ_c' and covariance Σ_c' are computed as follows:

$$\mu_c' = \Sigma_c' \left(\Sigma_c^{-1} \mu_c + \Sigma_{\text{new}}^{-1} \overline{\mathbf{X}}_c \right)$$

$$\Sigma_c' = \left(\Sigma_c^{-1} + \Sigma_{\text{new}}^{-1} \right)^{-1}$$

Here, Σ_{new} represents the covariance of the new data, and $\overline{\mathbf{X}}_c$ is the mean of the new features for class c. This Bayesian update process allows the model to refine its belief about each class's feature distribution over time.

The use of a Bayesian classifier in the code is intuitive primarily due to the probabilistic nature of the approach, which is well-suited for handling uncertainty in the data. Bayesian classifiers are based on Bayes' theorem, which allows us to make predictions by combining prior knowledge with observed data, and they are particularly effective when we need to model uncertainty or make predictions with incomplete information.

Pseudo-Labeling for Unlabeled Data

For the training procedure, the model utilizes both supervised learning (for labeled data) and pseudo-labeling (for unlabeled data). When dealing with the unlabeled datasets $D_2, D_3, \ldots D_{10}$, the model uses pseudo-labeling.

In pseudo-labeling, the classifier generates predictions for the unlabeled data based on its current posterior distributions. These predicted labels are accompanied by a confidence score. A threshold is applied to retain only the predictions with a high enough confidence. These high-confidence predictions are treated as pseudo-labels and used to update the posterior distribution.

The confidence threshold ensures that only the most reliable predictions are used for further training, helping the model refine its predictions even without access to true labels.

Model Evaluation

The model is evaluated on several held-out datasets, starting with D_1 and progressing through $D_2, D_3, \ldots D_{10}$. For each dataset, features are extracted using the ViT model, and the classifier predicts labels based on the learned posterior distributions. Accuracy is then computed by comparing the predicted labels with the true labels.

The accuracy for each dataset is tracked, allowing us to assess how well the model generalizes from the labeled training dataset D_1 to subsequent datasets with varying levels of available labels.

1.3 Results and experiments

Various other models such as updating mean in online manner, bayesian classifiers with different types of feature extractors such as ResNet and EfficientNet were tried out. Dimension reduction using kernalised PCAs etc were implemented and the best accuracy was given by was from the above-proposed model. The table below shows the accuracy of each model f_i trained on dataset D_j such that $i \leq j$. Matrix showing accuracy of each model across datasets

Table 1: Performance Metrics Across Features and Datasets

	$D1$	$D2$	$D3$	$D4$	$D5$	$D6$	$D7$	$D8$	$D9$	$D10$
$f1$	95.56%	0	0	0	0	0	0	0	0	0
$f2$	95.60%	96.16%	0	0	0	0	0	0	0	0
$f3$	95.56%	96.04%	95.56%	0	0	0	0	0	0	0
$f4$	95.56%	95.92%	95.48%	95.52%	0	0	0	0	0	0
$f5$	95.64%	95.92%	95.44%	95.48%	95.92%	0	0	0	0	0
$f6$	95.68%	95.84%	95.36%	95.48%	95.88%	96.00%	0	0	0	0
$f7$	95.60%	95.80%	95.24%	95.40%	95.88%	96.04%	95.40%	0	0	0
$f8$	95.60%	95.64%	95.24%	95.36%	95.88%	95.92%	95.36%	95.52%	0	0
$f9$	95.60%	95.64%	95.20%	95.36%	95.72%	95.80%	95.28%	95.44%	95.32%	0
$f10$	95.60%	95.56%	95.28%	95.40%	95.80%	95.80%	95.16%	95.44%	95.24%	96.20%

2 Task-1.2

Model Overview

The model employs a Vision Transformer (ViT) for feature extraction and utilizes two classifiers: a Bayesian classifier with Gaussian distributions and a mean update classifier. This system is designed to handle classification tasks, with sequential training on datasets 1-10 using a Gaussian method and updates on datasets 11-20.

For this task, we utilized the same model that was implemented in the previous task.

Bayesian Classifier (Gaussian Method)

The Bayesian classifier assumes each class has a Gaussian distribution, with a mean vector and covariance matrix. Given a dataset, the model computes posterior distributions using Bayes' theorem. The posterior distribution for each class is updated incrementally using the formula:

$$\mu_{\text{posterior}} = \left(\Sigma_{\text{prior}}^{-1} + \Sigma_{\text{likelihood}}^{-1}\right)^{-1} \left(\Sigma_{\text{prior}}^{-1}\mu_{\text{prior}} + \Sigma_{\text{likelihood}}^{-1}\mu_{\text{sample}}\right)$$

where: $\mu_{\text{posterior}}$ is the updated mean for a class, μ_{prior} is the prior mean, μ_{sample} is the sample mean from the current batch, Σ_{prior} and $\Sigma_{\text{likelihood}}$ are the prior and likelihood covariance matrices.

In the classification phase, the classifier computes the cosine similarity between the feature vector $\mathbf{f}$ and the class mean μ_c:

$$\text{cosine\_similarity} = \frac{\mathbf{f} \cdot \mu_{\mathbf{c}}}{|\mathbf{f}||\mu_{\mathbf{c}}|}$$

where $\mathbf{f}$ is the feature vector of the image, and $\mu_{\mathbf{c}}$ is the mean of class c.

Mean Update Classifier

In the second phase, the mean update classifier is introduced, which updates the class means incrementally as new data is processed. For each batch of data, the class means are updated by computing the running average of the feature vectors. The update for the class mean μ_c is given by:

$$\mu_c^{\text{new}} = \frac{n_c \mu_c^{\text{old}} + \mathbf{f}}{n_c + 1}$$

where n_c is the number of samples seen for class c so far, μ_c^{old} is the previous class mean, and $\mathbf{f}$ is the feature vector of the new sample.

The mean update classifier continues to update class means as more data becomes available, without the need to retrain from scratch. The cosine similarity between the feature vector $\mathbf{f}$ and the class mean μ_c is used for classification, as shown earlier.

Training and Evaluation Process

The training and evaluation process proceeds as follows:

- Initially, the model is trained on datasets 1-10 using the Bayesian classifier with the Gaussian method.
- After this, the model switches to the mean update classifier for incremental learning on datasets 11-20.
- The model's performance is evaluated based on its ability to correctly classify images from both previously seen datasets (D1 to D10) and new datasets (D11 to D20).
- During the evaluation phase, predictions are made by calculating the cosine similarity between the feature vector and the class means.

This approach enables the model to handle both old and new data, allowing for continuous updates to the classifier without retraining from scratch.

2.1 Results and experiments

Table 2: Performance Metrics Across Features and Datasets

	$D1$	$D2$	$D3$	$D4$	$D5$	$D6$	$D7$	$D8$	$D9$	$D10$
$f11$	95.04%	95.24%	95.04%	95.00%	95.24%	95.32%	94.96%	94.96%	95.44%	96.04%
$f12$	94.24%	94.36%	94.16%	94.24%	94.24%	94.84%	93.72%	94.32%	94.44%	95.00%
$f13$	95.04%	95.48%	94.76%	95.08%	94.96%	95.68%	94.80%	95.20%	95.40%	96.08%
$f14$	94.96%	95.44%	94.60%	94.84%	95.12%	95.48%	94.84%	94.92%	95.12%	95.88%
$f15$	95.04%	95.64%	95.40%	95.32%	95.72%	95.92%	95.32%	95.24%	95.52%	96.28%
$f16$	94.92%	95.60%	95.08%	95.24%	95.28%	95.52%	94.92%	95.08%	95.40%	96.40%
$f17$	94.32%	94.56%	94.20%	94.24%	94.56%	94.80%	94.56%	94.16%	94.64%	94.96%
$f18$	95.04%	95.44%	94.84%	95.20%	95.00%	95.44%	94.88%	94.96%	95.44%	95.96%
$f19$	94.28%	94.76%	93.88%	94.20%	93.88%	94.80%	94.36%	94.24%	94.40%	94.56%
$f20$	95.20%	95.20%	94.88%	95.08%	95.36%	95.20%	95.08%	95.08%	95.24%	95.84%

	$D11$	$D12$	$D13$	$D14$	$D15$	$D16$	$D17$	$D18$	$D19$	$D20$
$f11$	83.72%	0	0	0	0	0	0	0	0	0
$f12$	82.72%	77.44%	0	0	0	0	0	0	0	0
$f13$	83.64%	76.84%	88.64%	0	0	0	0	0	0	0
$f14$	83.36%	76.52%	88.80%	90.20%	0	0	0	0	0	0
$f15$	83.40%	76.92%	88.36%	90.80%	94.16%	0	0	0	0	0
$f16$	83.32%	76.24%	88.72%	90.32%	94.00%	86.72%	0	0	0	0
$f17$	82.32%	75.44%	87.76%	90.04%	93.28%	86.72%	77.32%	0	0	0
$f18$	82.96%	75.96%	88.48%	90.32%	94.12%	86.96%	79.04%	85.16%	0	0
$f19$	82.12%	74.36%	87.72%	89.40%	93.24%	85.28%	76.24%	83.92%	80.60%	0
$f20$	82.56%	76.24%	88.20%	90.40%	93.73%	86.96%	79.32%	85.04%	79.56%	90.44%

3 Task-2

The paper " Lifelong Domain Adaptation via Consolidated Internal Distribution" has been presented :
`https://youtu.be/07GpF8yrmPk`