CS771 - Intro to ML (Autumn 2024): Mini-project 2

Kushagra Srivastava
24111041
Kumari Ritika
241110039
Suvam Mukhopadhyay
241110074
Shubhashish Shukla
241110069
Krishanu Ray
241110037

November 26, 2024

Contents

1 Problem 1

1.1 Introduction

This section describes the task of training models using labeled and unlabeled subsets of CIFAR-10 with the goal of achieving good accuracy while maintaining stability across datasets. This code implements a continual learning pipeline using a pre-trained ConvNeXt Small feature extractor and Learning with Prototypes(LwP) classifier for image classification tasks. To execute the code as intended, first execute Task_1.ipynb and then Task_2.ipynb to get proper outputs.

1.2 Task 1

The code has the following main stages:

1.2.1 Feature Extraction

- A pre-trained model called ConvNeXt Small is loaded with pretrained weights.
- The classification head(final layer) is removed to retain only the feature extraction layers.
- Global Average Pooling is applied to obtain compact feature vectors which has a size of 786.

1.2.2 Data Transformations

- Training and evaluation datasets are loaded dynamically in a loop.
- Images are preprocessed with resizing, normalization, and augmentation using PyTorch transforms to meet ConvNeXt input requirements.
- A custom dataset class (UnlabeledDataset) facilitates loading of data in batches of defined batch size.

1.2.3 Classification using Learning with Prototypes

- A prototype-based classifier is defined, where class prototypes are the mean of feature vectors for each class.
- Predictions are made by assigning each feature to the nearest prototype in feature space.

1.2.4 Main loop

In each loop, the following takes place:

- Features of the training set are extracted and prototypes are initialized.
- The classifier is trained using these features and corresponding labels.
- Features from all held-out datasets seen so far are evaluated for accuracy.

Features of the next dataset are pseudo-labeled using the current model and used for subsequent training. To prevent re-computations, features extracted from the next dataset and held-out datasets are stored.

1.2.5 Performance Tracking and storing the model $\mathbf{f}_{10}$

- An accuracy matrix (acc_matrix) records performance for all tasks.
- The trained classifier is saved as a serialized pickle file (lwp_classifier.pkl) after the last task using Pickle module.

1.2.6 Results for Task 1

The performance of each model $f_1, \ldots, f_{10}$ on the held-out datasets $\hat{D}_1, \ldots, \hat{D}_{10}$ is summarized in Table 1.

Model	$\hat{D}_1$	$\hat{D}_2$	$\hat{D}_3$	$\hat{D}_4$	$\hat{D}_5$	$\hat{D}_6$	$\hat{D}_7$	$\hat{D}_8$	$\hat{D}_9$	$\hat{D}_{10}$
f_1	93.16	-	-	-	-	-	-	-	-	-
f_2	92.68	93.00	-	-	-	-	-	-	-	-
f_3	92.40	92.56	93.04	-	-	-	-	-	-	-
f_4	92.40	92.52	93.20	93.12	-	-	-	-	-	-
f_5	92.28	92.84	93.40	93.12	93.40	-	-	-	-	-
f_6	92.16	92.48	92.96	92.48	92.80	92.92	-	-	-	-
f_7	91.44	92.04	92.32	92.28	92.16	92.24	93.20	-	-	-
f_8	91.80	91.96	92.32	92.16	92.44	92.60	93.12	92.08	-	-
f_9	92.08	91.92	92.36	92.32	92.60	92.52	93.44	92.28	92.40	-
f_{10}	91.72	91.72	92.00	91.80	91.96	91.92	92.84	91.76	91.64	92.72

Table 1: Accuracy Matrix for Task 1

1.2.7 Observations

- We notice that the accuracies of every model are consistent throughout the datasets.
- This shows that the datasets $\hat{D}_1, \ldots, \hat{D}_{10}$ inputs are derived from the same distribution, as stated in the problem statement.

1.3 Task 2

This code builds upon the previous implementation of a continual learning pipeline using the same pre-trained ConvNeXt feature extractor and a prototype-based classifier (LWPClassifier). However, it introduces new datasets and a different workflow to leverage previously trained classifiers for pseudo-labeling. The previously trained LWPClassifier f_{10} is loaded using Python's pickle library and used to train further models.

1.3.1 Feature Extraction

- Same as Task 1, ConvNeXt-Small is utilized for feature extraction.
- The classification head is removed, and Global Average Pooling (GAP) is applied to obtain compact feature vectors of size 786.

1.3.2 Data Transformations

- Training and evaluation datasets are loaded dynamically in a loop.
- Datasets are transformed to meet ConvNeXt input requirements: Images resized to 224×224, normalized, and tensorized.
- A custom dataset class (UnlabeledDataset) facilitates loading of data in batches of defined batch size.

1.3.3 Main Loop

Each iteration has the following stages:

1. Training Features:
    - Extract features from training images using the feature extractor.
    - Predict pseudo-labels for the training set using the loaded LWPClassifier.
    - Train the classifier on extracted features and pseudo-labels.
2. Evaluating Features:
    - Extract features from evaluation images.
    - Predict labels using the updated classifier.
    - Record accuracy for all previously seen tasks in the accuracy matrix (acc_matrix).

1.3.4 Results for Task 2

For Task 2, the models $f_{11}, \ldots, f_{20}$ were trained on datasets $D_{11}, \ldots, D_{20}$ considering domain shifts. The results are shown in Table 2.

Model	$\hat{D_{11}}$	$\hat{D_{12}}$	$\hat{D_{13}}$	$\hat{D_{14}}$	$\hat{D_{15}}$	$\hat{D_{16}}$	$\hat{D_{17}}$	$\hat{D_{18}}$	$\hat{D_{19}}$	$\hat{D_{20}}$
f_{11}	76.84	-	-	-	-	-	-	-	-	-
f_{12}	73.36	62.76	-	-	-	-	-	-	-	-
f_{13}	74.76	60.64	81.92	-	-	-	-	-	-	-
f_{14}	76.16	60.08	82.44	90.32	-	-	-	-	-	-
f_{15}	75.08	59.28	82.32	90.36	90.56	-	-	-	-	-
f_{16}	74.88	59.12	82.44	88.64	89.56	76.88	-	-	-	-
f_{17}	74.16	58.80	81.24	88.36	88.72	77.08	82.64	-	-	-
f_{18}	74.64	59.40	82.32	88.00	88.60	75.92	82.68	79.48	-	-
f_{19}	72.36	57.52	80.00	85.96	86.60	73.84	78.40	78.28	75.68	-
f_{20}	74.08	57.56	81.20	87.56	88.56	75.52	80.00	79.44	74.24	86.60

Table 2: Accuracy Matrix for Task 2

1.3.5 Observations

- We notice that the accuracies of every model varies greatly across the datasets.
- This shows that the datasets $\hat{D_{11}}, \ldots, \hat{D_{20}}$ inputs are derived from different distributions, as stated in the problem statement.

2 Problem 2: Paper Presentation

2.1 Chosen Paper:

DEJA VU: CONTINUAL MODEL GENERALIZATION FOR UNSEEN DOMAINS

- **Problem Studied:** This paper addresses the problem of how deep learning models can adapt to continually changing, unseen data distributions in real-world environments.
- **Key ideas proposed:** The authors propose RaTP, a novel framework aimed at achieving three objectives:
  1. Target Domain Generalization (TDG): Enhance model performance on new domains before any adaptation occurs.
  2. Target Domain Adaptation (TDA): Facilitate rapid and effective adaptation once new domains are encountered.
  3. Forgetting Alleviation (FA): Mitigate the loss of previously learned knowledge as domains shift.
- **Results:** Extensive experiments on datasets like Digits, PACS, and DomainNet demonstrate that RaTP significantly outperforms state-of-the-art methods across TDG, TDA, and FA metrics. It achieves better generalization during the Unfamiliar Period while remaining competitive in domain adaptation and memory retention.

2.2 Presentation Link

The video presentation summarizing the chosen paper is available at the following link:
`https://www.youtube.com/watch?v=NX9mY7_sheo`

* * *