HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

Overview of Continual Learning and Application in Defect Classification

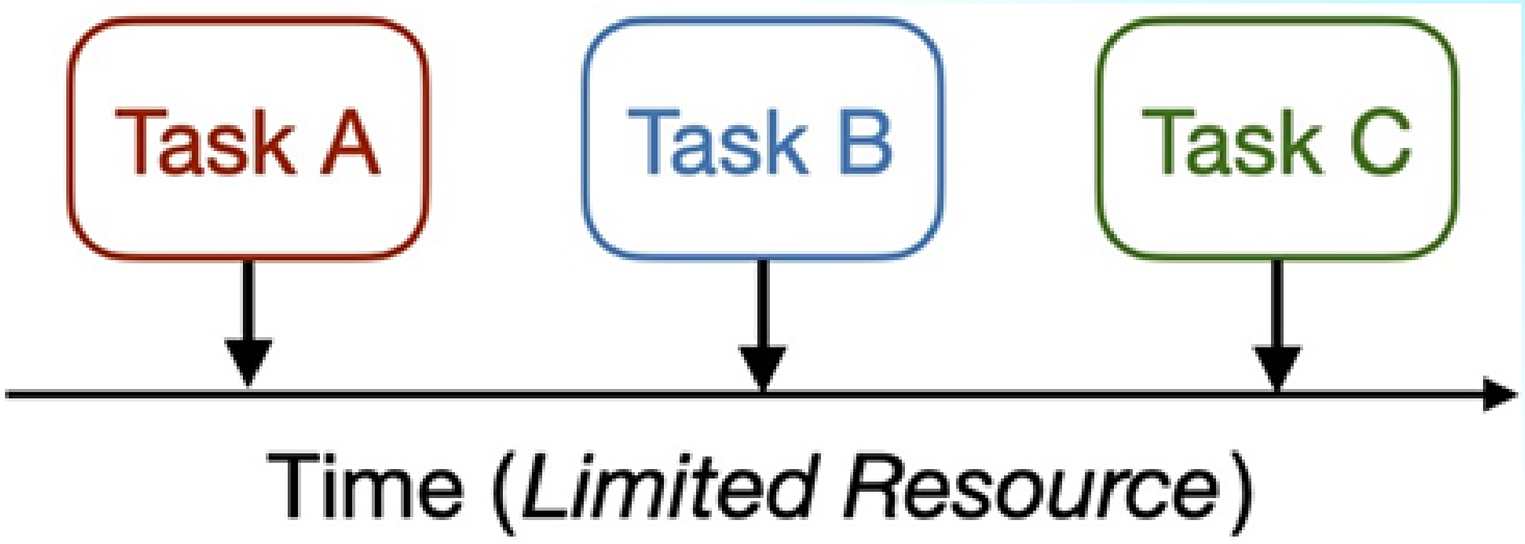
Presenter: La Nguyen Gia Hy

Content

Overview of Continual Learning

Definition

Continual Learning (also known as **Incremental Learning**, **Life-long Learning**) is a concept to learn a model for a large number of tasks sequentially without forgetting knowledge obtained from the preceding tasks, where the data in the old tasks are not available anymore during training new ones



Overview of Continual Learning

Typical Scenarios

- **Instance-Incremental Learning (IIL):** All training samples belong to the same task and arrive in batches.
- **Domain-Incremental Learning (DIL):** Tasks have the same data label space but different input distributions.
- **Task-Incremental Learning (TIL):** Tasks have disjoint data label spaces. Task identities are provided in both training and testing.
- **Class-Incremental Learning (CIL):** Tasks have disjoint data label spaces. Task identities are only provided in training.

.

Overview of Continual Learning

Evaluation metrics

- **Average Accuracy (AA):** Let $a_{k,j} \in [0,1]$ denote the classification accuracy evaluated on the test set of the j-th task after incremental learning of the k-th task (j ≤ k).

$$AA_k = \frac{1}{k}\sum_{j=1}^{k} a_{k,j}$$

- **Backward Transfer (BWT)**: evaluates the average influence of learning the k-th task on all old tasks, indicating **memory stability.**

$$BWT_k = \frac{1}{k-1}\sum_{j=1}^{k-1}(a_{k,j} - a_{j,j})$$

- **Forward Transfer (FWT):** evaluates the average influence of all old tasks on the current k-th task, indicating **learning plasticity**.

$$FWT_k = \frac{1}{k-1}\sum_{j=2}^{k}(a_{j,j} - \tilde{a}_j)$$

$\tilde{a}_j$:classification accuracy of a randomly-initialized reference model trained with j-th task

Overview of Continual Learning

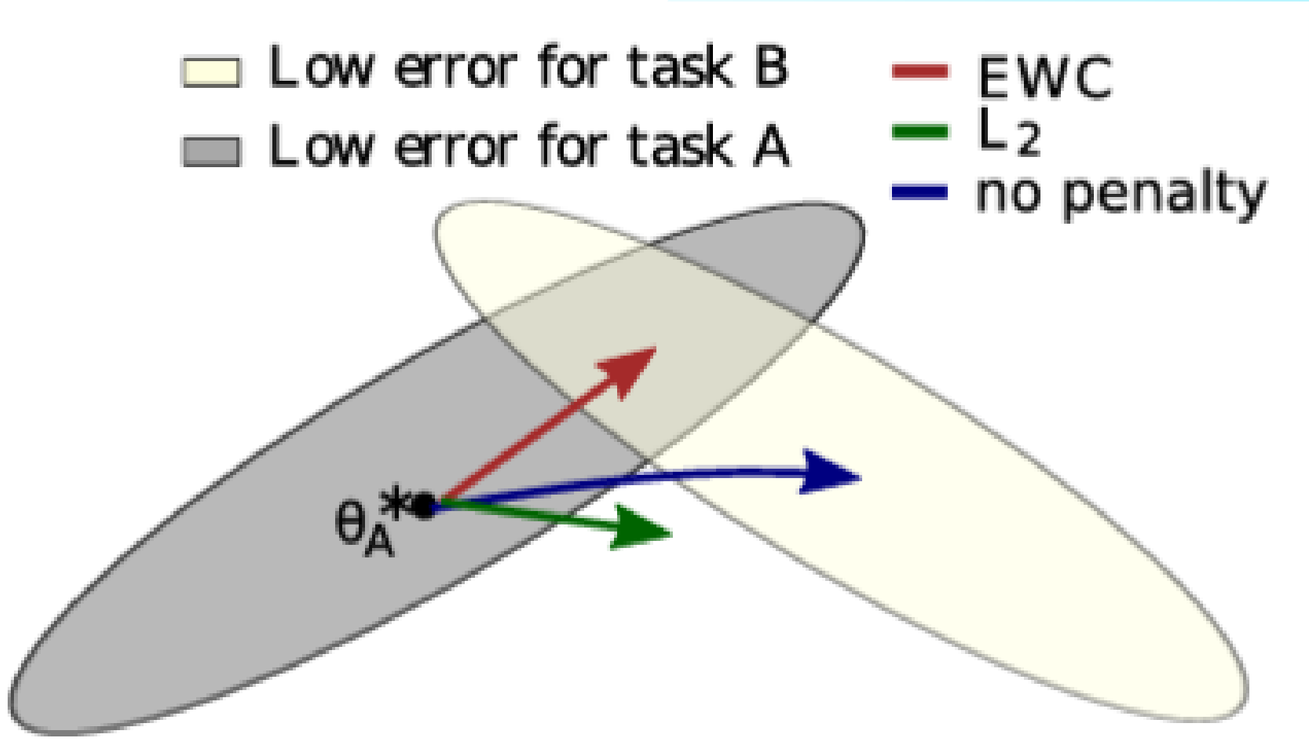
Main Categories of CL methods

- **Regularization-based approach:** adding explicit regularization terms to balance the old and new tasks, usually requires storing a frozen copy of the old model for reference. Example: LwF, EWC
- **Replay-based approach:** approximating and recovering old data distributions by stores a few old training samples within a small memory buffer. Example: iCaRL
- **Optimization-based approach:** explicitly designing and manipulating the optimization programs. Example: GEM, A-GEM
- **Representation-based approach:** create and exploit the strengths of representations for continual learning, either by self-supervised learning or use pre-training for downstream CL.
- **Architecture-based approach:** constructing task-specific parameters to explicitly resolve inter-task interference

Elastic Weight Consolidation (EWC)

Key idea: Protect important weights from changing too much when learning new tasks.

- Uses Fisher Information Matrix to identify crucial parameters.
- Applies stronger regularization to those weights.
- Allows other weights to adapt to the new task



Kirkpatrick, James, et al. "Overcoming catastrophic forgetting in neural networks." (2017)

How EWC Works

1. Train on Task A and compute Fisher Information matrix to **identify important weights**.
2. When learning Task B, EWC **apply a penalty** to changes in those weights.

$$L = L_{new\,task} + \lambda \sum_i F_i(\theta_i - \theta_i^*)^2$$

- F_i is the Fisher Information (importance of weight)
- θ_i^* is the previous weight value
- λ controls the strength of regularization

Kirkpatrick, James, et al. "Overcoming catastrophic forgetting in neural networks." (2017)

Incremental Classfier and Representation Learning (iCaRL)

Overview

Internal representation:

- feature function $\varphi : \mathcal{X} \to \mathbb{R}^d$, weight vectors w_y for $y \in \mathcal{Y}$
- for each seen class, y, a set of exemplar samples, P_y (in total up to K samples)

For representation learning:

- probabilistic outputs: $g_y(x) = \frac{1}{1 + e^{-\langle w_y, \varphi(x) \rangle}}$ for each $y \in \mathcal{Y}$ seen so far
- Loss function consists of classification and distillation terms.

For classification:

- classify samples by their distance to a class prototype (like nearest-class-mean), but using the mean of exemplars, not all training examples

Rebuffi et al. "icarl: Incremental classifier and representation learning, CVPR 2017

Incremental Classfier and Representation Learning (iCaRL)

Incremental Training

IncrementalTrain$(X^s, \ldots, X^t, K)$

input $X^s, \ldots, X^t$ // new training examples in per-class sets (all $x \in X^y$ are of class y)
input K // maximum number of exemplars
require Θ // current model parameters
require $\mathcal{P} = (P_1, \ldots, P_{s-1})$ // current exemplar sets
$\Theta \leftarrow$ **UpdateRepresentation**$(X^s, \ldots, X^t; \mathcal{P}, \Theta)$
$m \leftarrow \lfloor K/t \rfloor$ // number of exemplars per class
for $y = 1, \ldots, s-1$ **do**
$P_y \leftarrow$ **ReduceExemplarSet**(P_y, m)
end for
for $y = s, \ldots, t$ **do**
$P_y \leftarrow$ **ConstructExemplarSet**(X_y, m, Θ)
end for
$\mathcal{P} \leftarrow (P_1, \ldots, P_t)$ // new exemplar sets

Incremental Classfier and Representation Learning (iCaRL)

Update Representation

The training set:

$$\mathcal{D} \leftarrow \bigcup_{y=s,\ldots,t} \{(x,y) : x \in X^y\} \cup \bigcup_{y=1,\ldots,s-1} \{(x,y) : x \in P^y\}$$

consists of samples of new classes, but also exemplars of old classes

→ representation is ‘reminded’ of old classes regularly

The loss function:

$$\ell(\Theta) = -\sum_{(x_i,y_i)\in\mathcal{D}} [\underbrace{\sum_{y=s}^{t} \delta_{y=y_i} \log(g_y(x_i))}_{\text{classification loss}} + \underbrace{\sum_{y=1}^{s-1} q_i^y \log\left(g_y(x_i)\right)}_{\text{distillation loss}}]$$

contains not just ordinary classification term, but also *distillation* term

Rebuffi et al. "icarl: Incremental classifier and representation learning, CVPR 2017

Incremental Classfier and Representation Learning (iCaRL)

Exemplar Management

$P \leftarrow \text{REDUCEEXEMPLARSET}(P, m)$

input $P = (p_1, \ldots, p_{|P|})$ // current exemplar set
input m // target number of exemplars

output exemplar set $P = (p_1, \ldots, p_m)$ // keep only first m exemplars

$P \leftarrow \text{CONSTRUCTEXEMPLARSET}(X, m)$

input sample set $X = \{x_1, \ldots, x_n\}$ of class y
input m target number of exemplars
require current feature function $\varphi : \mathcal{X} \to \mathbb{R}^d$

$\mu \leftarrow \frac{1}{n} \sum_{x \in X} \varphi(x)$ // current class mean
for $k = 1, \ldots, m$ **do**
$p_k \leftarrow \underset{x \in X}{\text{argmin}} \left\| \mu - \frac{1}{k} [\varphi(x) + \sum_{j=1}^{k-1} \varphi(p_j)] \right\|$ // next exemplar
end for

output exemplar set $P = (p_1, \ldots, p_m)$

Incremental Classfier and Representation Learning (iCaRL)

Classification

$y^* \leftarrow \textsc{NearestMeanOfExemplars}(x)$

input x // sample to be classified

require $\mathcal{P} = (P_1, \ldots, P_t)$ // class exemplar sets of classes $1, \ldots, t$ seen so far

require $\varphi : \mathcal{X} \to \mathbb{R}^d$ // feature map

for $y = 1, \ldots, t$ **do**

$$\mu_y \leftarrow \frac{1}{|P_y|} \sum_{p \in P_y} \varphi(p) \qquad \text{// mean-of-exemplars}$$

end for

$$y^* \leftarrow \underset{y=1,\ldots,t}{\operatorname{argmin}} \|\varphi(x) - \mu_y\| \qquad \text{// nearest prototype}$$

output class label y^*

Rebuffi et al. "icarl: Incremental classifier and representation learning, CVPR 2017

Contrastive Continual Learning

Motivation

Instead of asking how to isolate previous knowledge from new knowledge, we draw attention to the following fundamental question:

What type of knowledge is likely to be useful for future tasks (and thus not get forgotten), and how can we learn and preserve such knowledge?

- Consider the simple scenario of classifying the given image as an **apple** or a **banana**
- An easy way to solve this problem is to use the **color feature**
- But it will no longer be useful if our future task is to classify another set of images as **apples** or **strawberries**
- If the model had learned more complicated features, e.g., **shape/polish/texture**, the features may be re-used for future tasks and remain unforgotten.

Cha, Hyuntak, Jaeho Lee, and Jinwoo Shin. "Co2l: Contrastive continual learning." ICCV 2021.

Contrastive Continual Learning

Motivation

To learn more transferable representations for continual learning, we can use *contrastive learning* - a recent advance in self-supervised learning.

However, applying this idea to continual settings is not straightforward:

1. The instantaneous demographics of **negative samples** are severely restricted under standard continual setups
   In class-incremental learning, the learner can access samples from only a small number of classes at each time step
2. How to preserve the learned *representations* **not** as a part of a **jointly trained** representation-classifier pair?
   Recent works on representation learning for continual setups lack an explicit design for a decoupled learning setup

Cha, Hyuntak, Jaeho Lee, and Jinwoo Shin. "Co2l: Contrastive continual learning." ICCV 2021.

Contrastive Continual Learning

Preliminaries: Supervised Contrastive Learning

Suppose that the classification model can be decomposed into two components: $\varphi_\theta = \mathbf{w} \circ f_\vartheta$ with parameter pairs $\theta = (\mathbf{w}, \vartheta)$, where:

- $\mathbf{w}(\cdot)$ is the **linear classifier**,
- $f_\vartheta(\cdot)$ is the representation (**encoder**)

Given a batch of N training samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, SupCon first generates an augmented batch $\{(\tilde{\mathbf{x}}_i, \tilde{y}_i)\}_{i=1}^{2N}$ by making a two randomly augmented versions of $\mathbf{x}_k$ as $\tilde{\mathbf{x}}_{2k}, \tilde{\mathbf{x}}_{2k-1}$

The samples in the augmented batch are mapped to a unit d-dimensional Euclidean sphere $\mathbf{z}_i = (g \circ f)_\psi(\tilde{\mathbf{x}}_i)$, where:

- $g = g_\phi$ denotes the **projection map** parametrized by ϕ
- ψ denotes the concatenation of ϑ and ϕ

Cha, Hyuntak, Jaeho Lee, and Jinwoo Shin. "Co2l: Contrastive continual learning." ICCV 2021.

Contrastive Continual Learning

Preliminaries: Supervised Contrastive Learning

The feature map $(g \circ f)_\psi$ is trained to minimize the supervised contrastive loss

$$\mathcal{L}^{sup} = \sum_{i=1}^{2N} \frac{-1}{|\mathfrak{p}_i|} \sum_{j \in \mathfrak{p}_i} \log \left(\frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{\sum_{k \neq i} \exp\left(\mathbf{z}_i \cdot \mathbf{z}_k / \tau\right)} \right)$$

Where $\tau > 0$ is some temperature hyperparameter and $\mathfrak{p}_i$ is the index set of positive samples with respect to the anchor

$$\mathfrak{p}_i = \{j \in \{1, \ldots, 2N\} | j \neq i, y_j = y_i\}$$

In other words, the sample in $\mathfrak{p}_i$ is either the other augmentation of the unaugmented version of $\tilde{\mathbf{x}}_i$, or one of the other augmented samples having the same label.

Cha, Hyuntak, Jaeho Lee, and Jinwoo Shin. "Co2l: Contrastive continual learning." ICCV 2021.

Contrastive Continual Learning

Main method - Asymmetric SupCon Loss

At a high level, Co2L *learns* the representations with (1) an **asymmetric form of supervised contrastive loss** and (2) *preserves* learned representations using **self-supervised distillation** in a decoupled representation-classifier training scheme

$$\mathcal{L} = \underbrace{\mathcal{L}_{asym}^{sup}}_{(1)\ \text{learning}} + \underbrace{\lambda \cdot \mathcal{L}^{IRD}}_{(2)\ \text{preserving}}$$

Let $S \subset \{1, \ldots, 2N\}$ be the set of indices of current task samples in the batch, the modified supervised contrastive loss is defined as

$$\mathcal{L}_{asym}^{sup} = \sum_{i \in S} \frac{-1}{|\mathfrak{p}_i|} \sum_{p \in \mathfrak{p}_i} \log \left(\frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau)}{\sum_{k \neq i} \exp\left(\mathbf{z}_i \cdot \mathbf{z}_k / \tau\right)} \right)$$

This asymmetric design is to prevent a model from **overfitting** to a small number of **past task samples.**

Cha, Hyuntak, Jaeho Lee, and Jinwoo Shin. "Co2l: Contrastive continual learning." ICCV 2021.

Contrastive Continual Learning

Main method - Instance-wise Relation Distillation (IRD)

For each sample $\tilde{\mathbf{x}}_i$ in a batch $\mathcal{B}$, we define the *instance-wise similarity vector*:

$$\mathbf{p}(\tilde{\mathbf{x}}_i; \psi; \kappa) = [p_{i,1}, \ldots, p_{i,i-1}, p_{i,i+1}, \ldots, p_{i,2N}]$$

where $p_{i,j}$ denotes the normalized instance-wise similarity

$$p_{i,j} = \frac{\exp\left(\mathbf{z}_i \cdot \mathbf{z}_j / \kappa\right)}{\sum_{k \neq i}^{2N} \exp\left(\mathbf{z}_i \cdot \mathbf{z}_k / \kappa\right)}$$

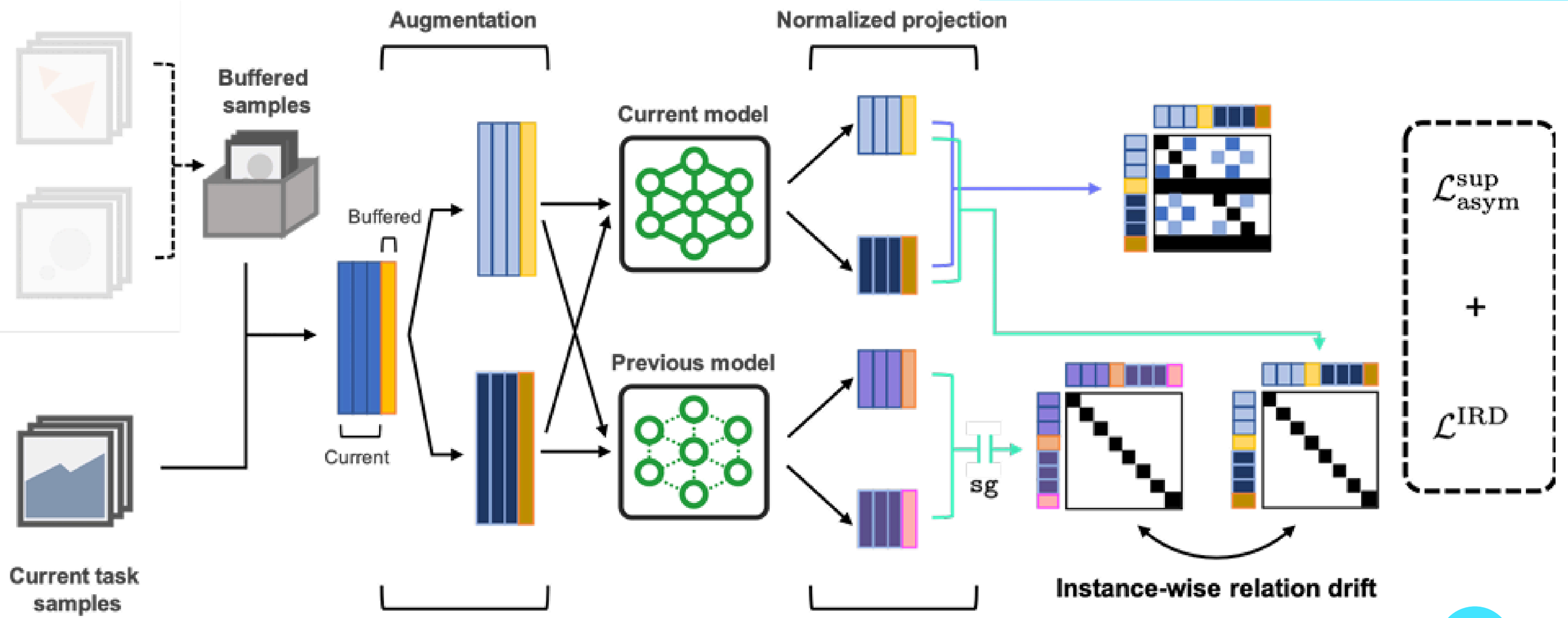
Roughly, the IRD loss quantifies the discrepancy between the instance-wise similarities of the current representation and the past representation (a snapshot of the model at the end of previous task):

$$\mathcal{L}^{IRD} = \sum_{i=1}^{2N} -\mathbf{p}(\tilde{\mathbf{x}}_i; \psi^{past}, \kappa^*) \cdot \log \mathbf{p}(\tilde{\mathbf{x}}_i; \psi, \kappa)$$

→ IRD **distills learned representations** to the current training model, thereby leading to *preserving* learned representations.

Contrastive Continual Learning

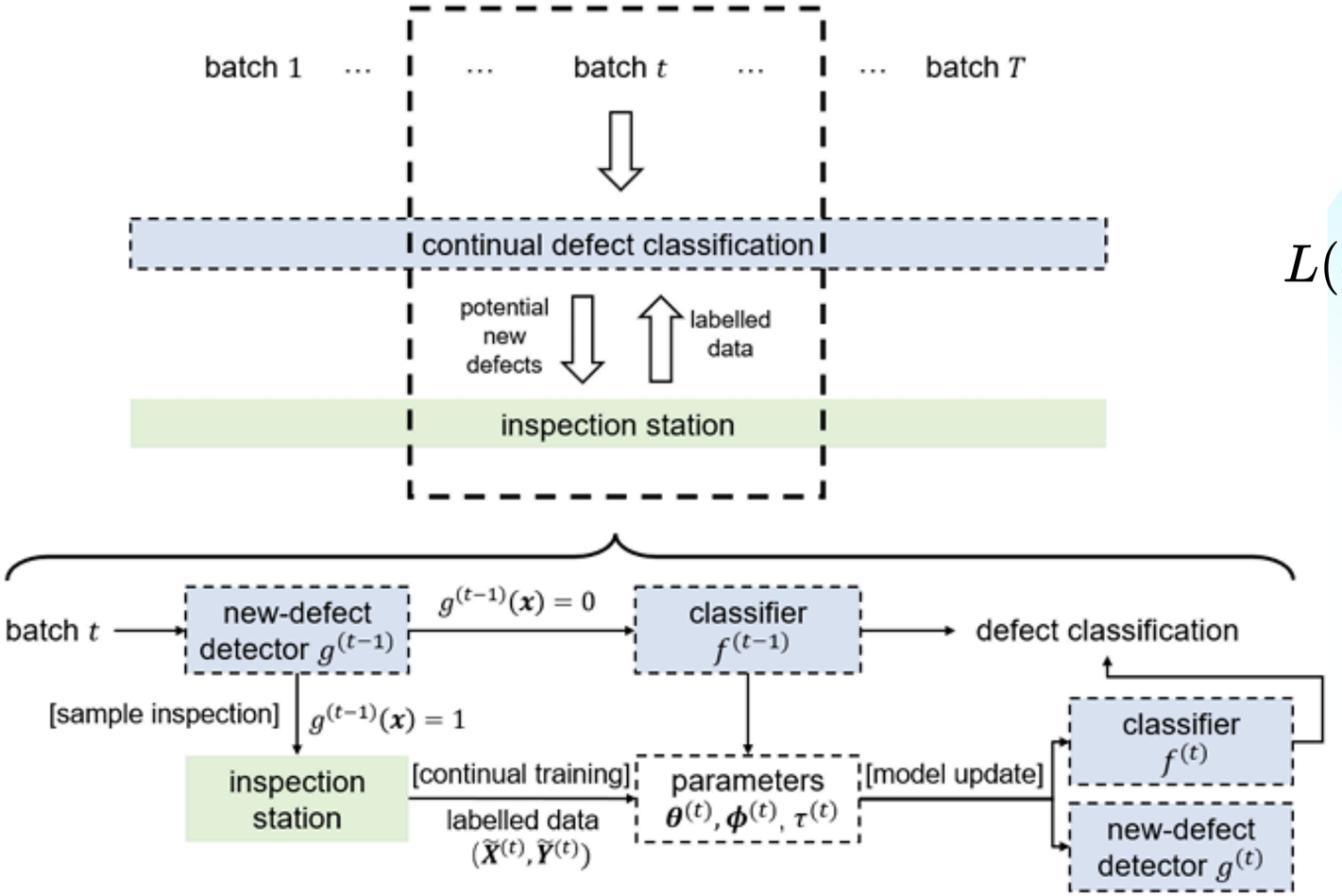
Main method



Cha, Hyuntak, Jaeho Lee, and Jinwoo Shin. "Co2l: Contrastive continual learning." ICCV 2021.

Implementation of CL in Defect Classification

Framework



Loss function

$$L(\theta, \phi) = L_{class}(\theta) + \lambda_{prior} L_{prior}(\theta) + \lambda_{ood} L_{ood}(\phi)$$

L_{class} Standard CE loss on current labelled data.

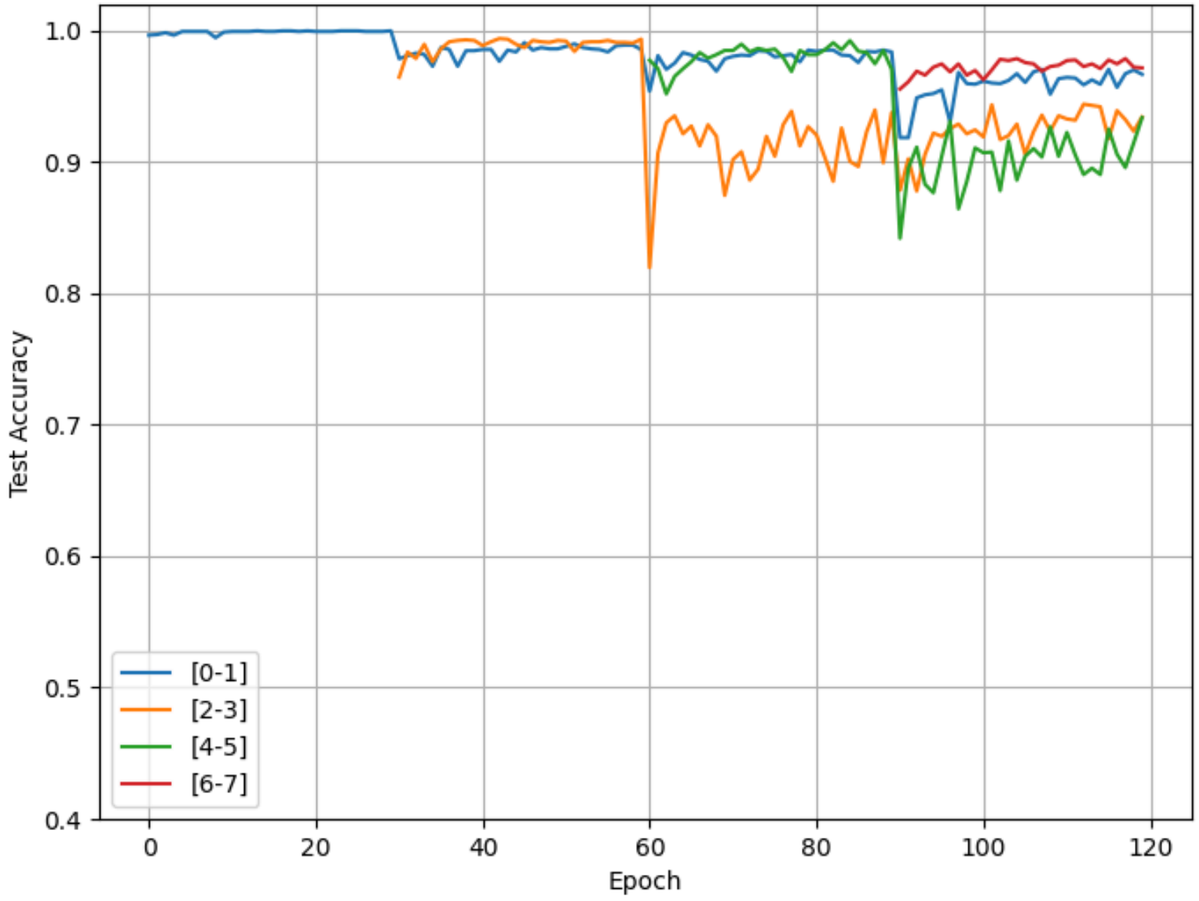
L_{prior} Regularization using Fisher information matrix to preserve prior learning

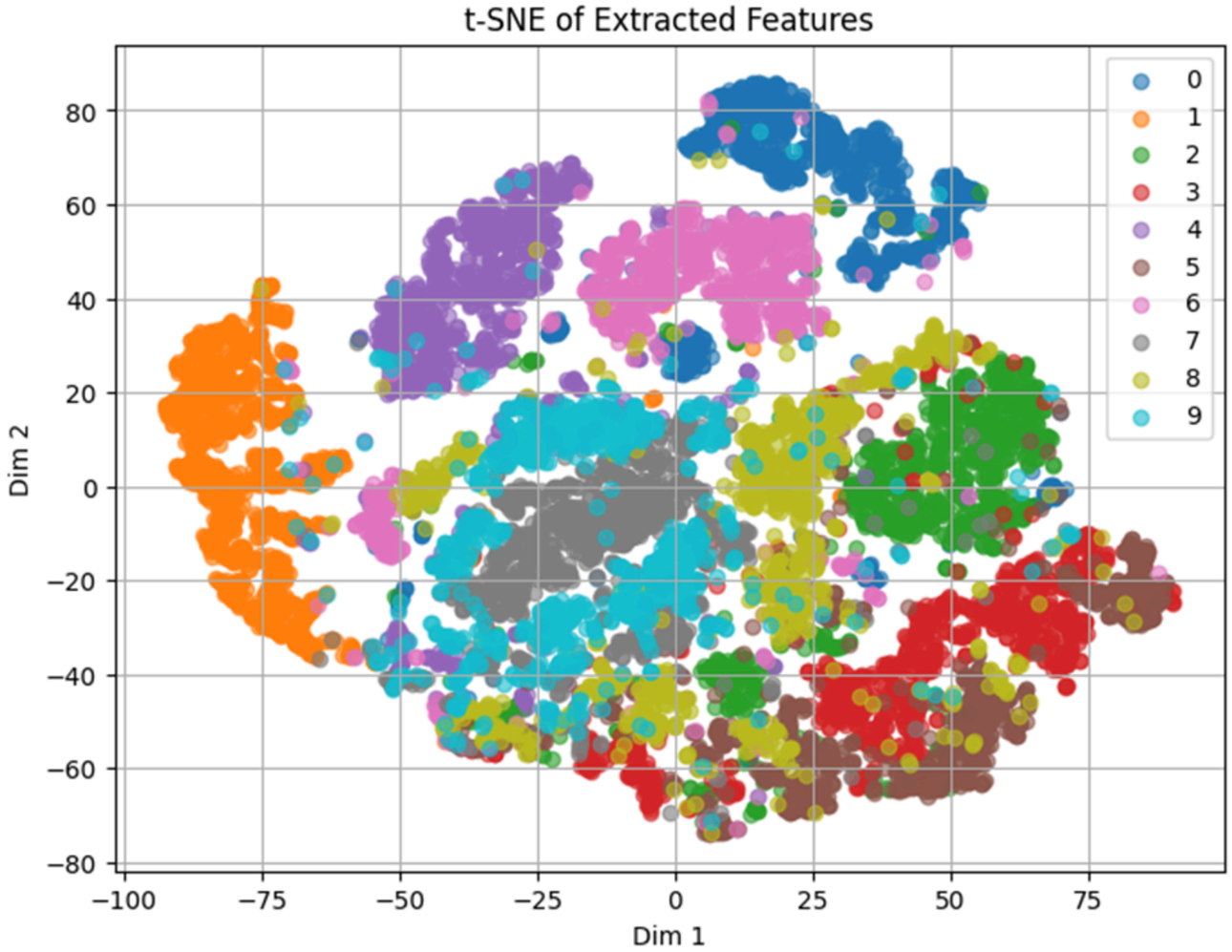
L_{ood} Penalizes incorrect classification of OOD vs in-distribution samples.

Sun, et al. "A continual learning framework for adaptive defect classification and inspection." Journal of Quality Technology (2023)

Visualization for MNIST

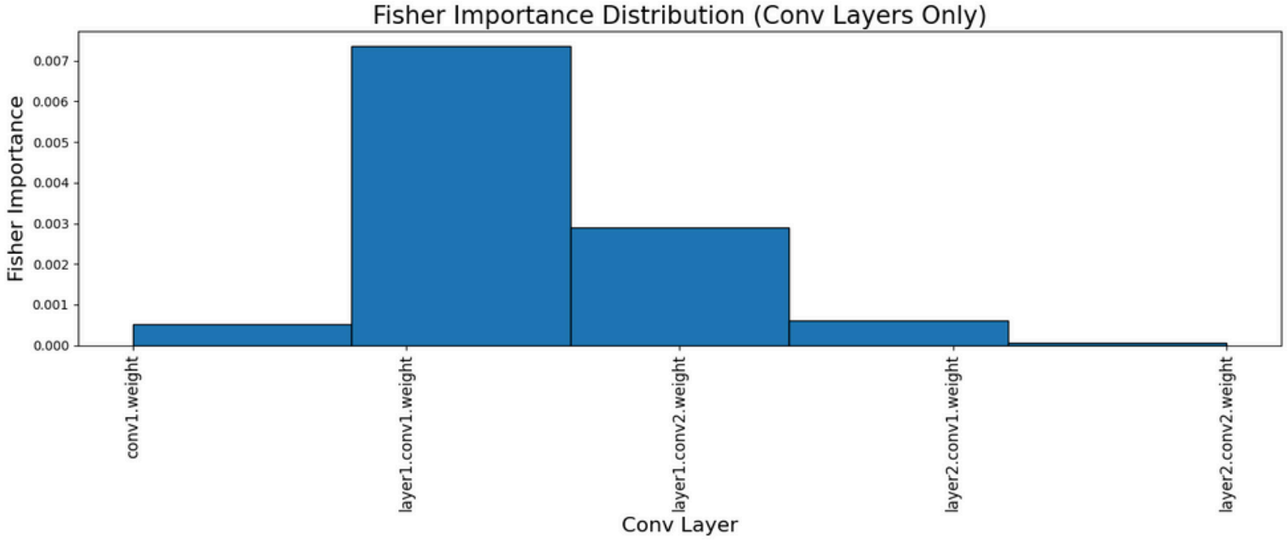
- Architecture: Small ResNet with 2 layers, each with 2 conv blocks
- 4 tasks, 2 classes per task, 30 epochs per task
- Average accuracy: 94.69%

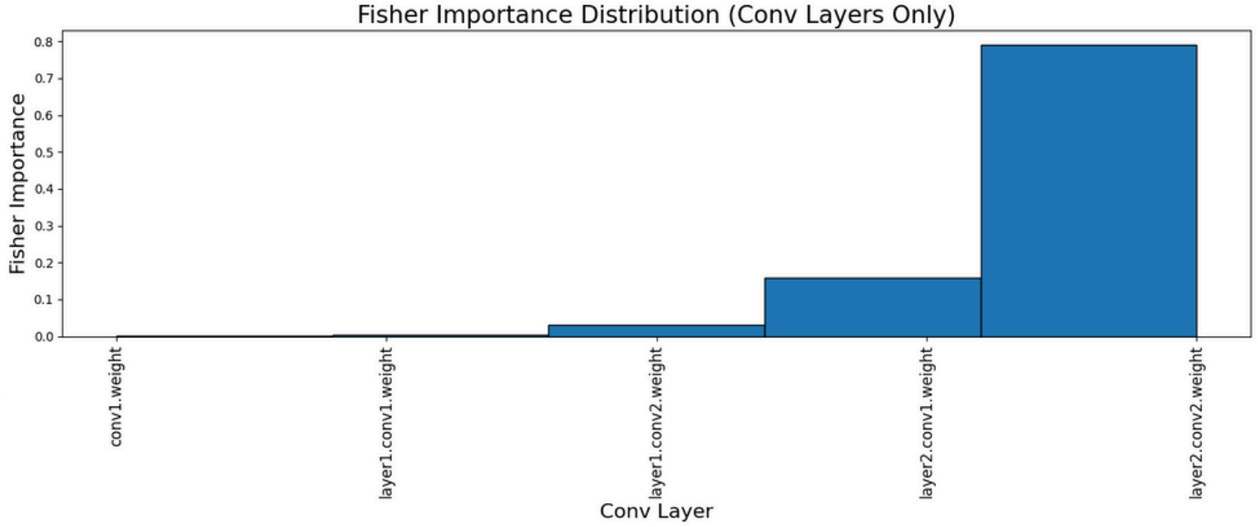


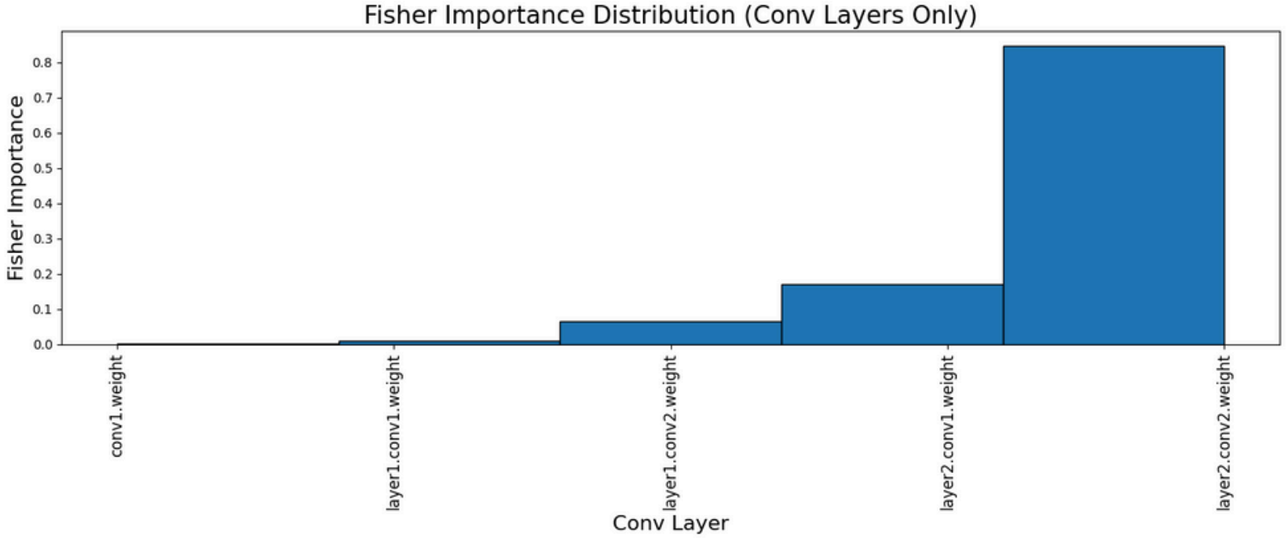


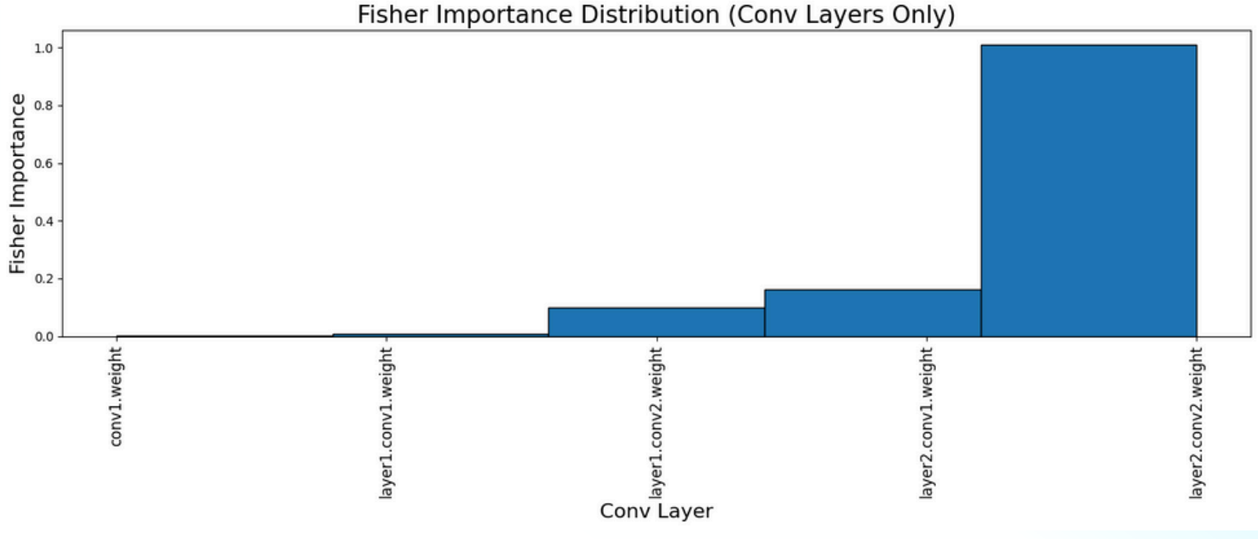
Visualization for MNIST

- Fisher information importance by conv layers after each task



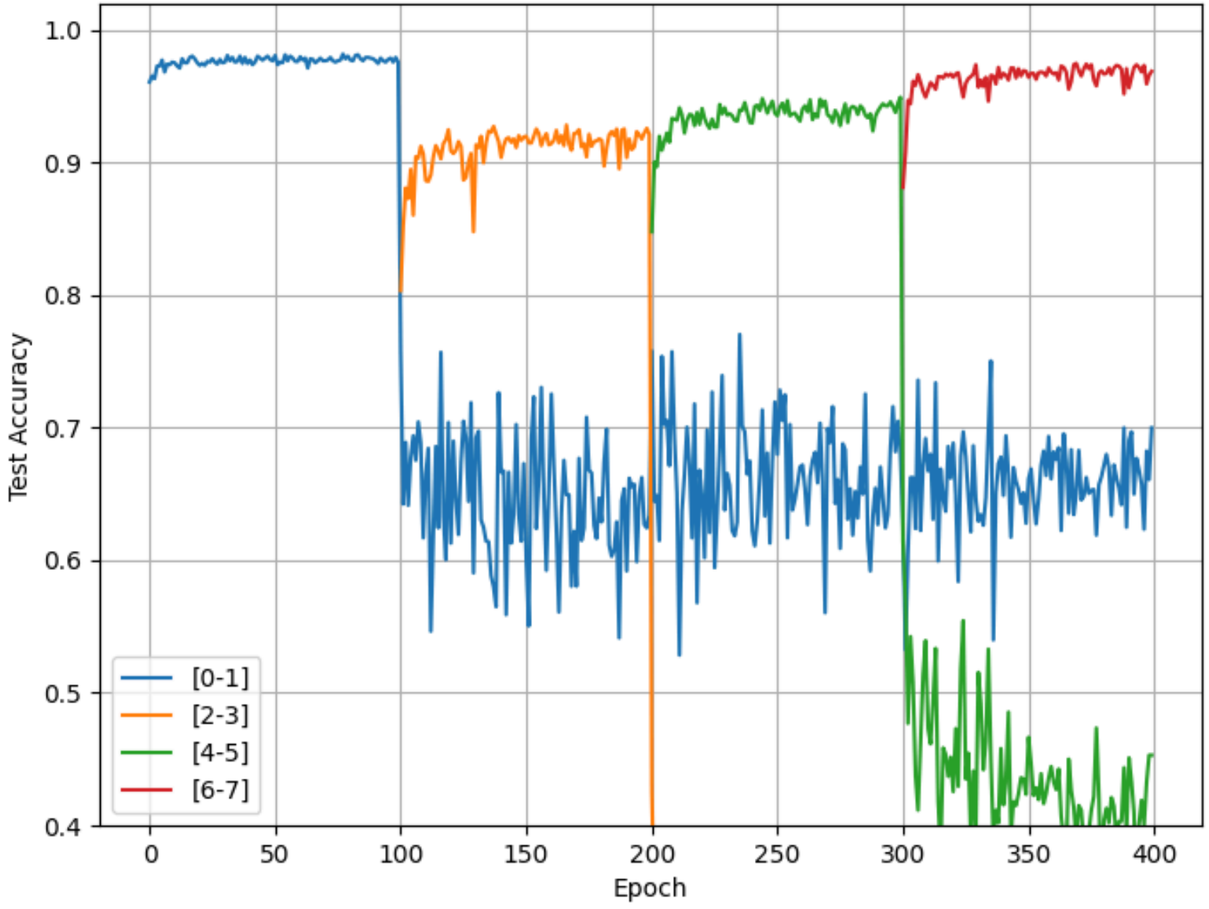


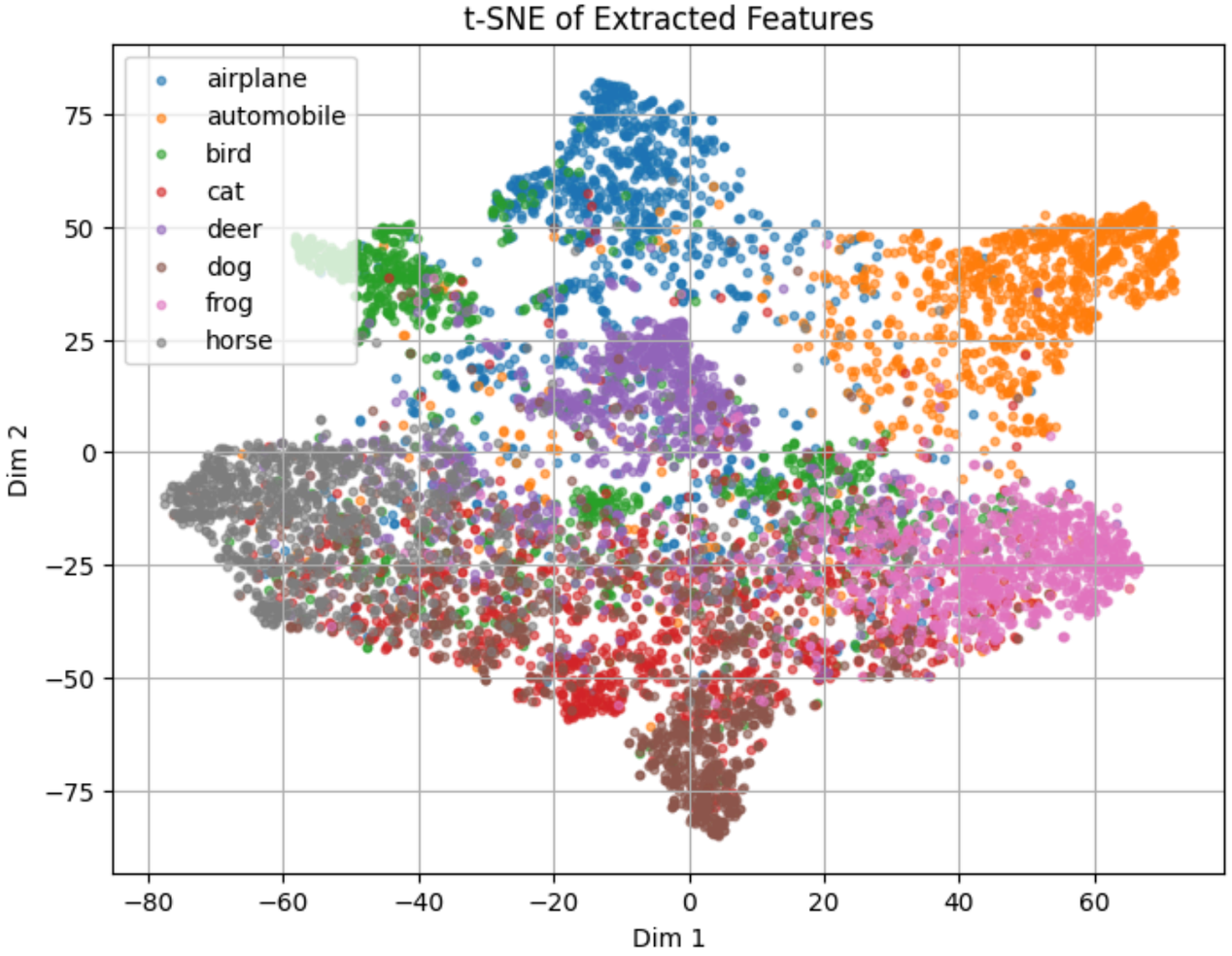




Visualization for CIFAR-10

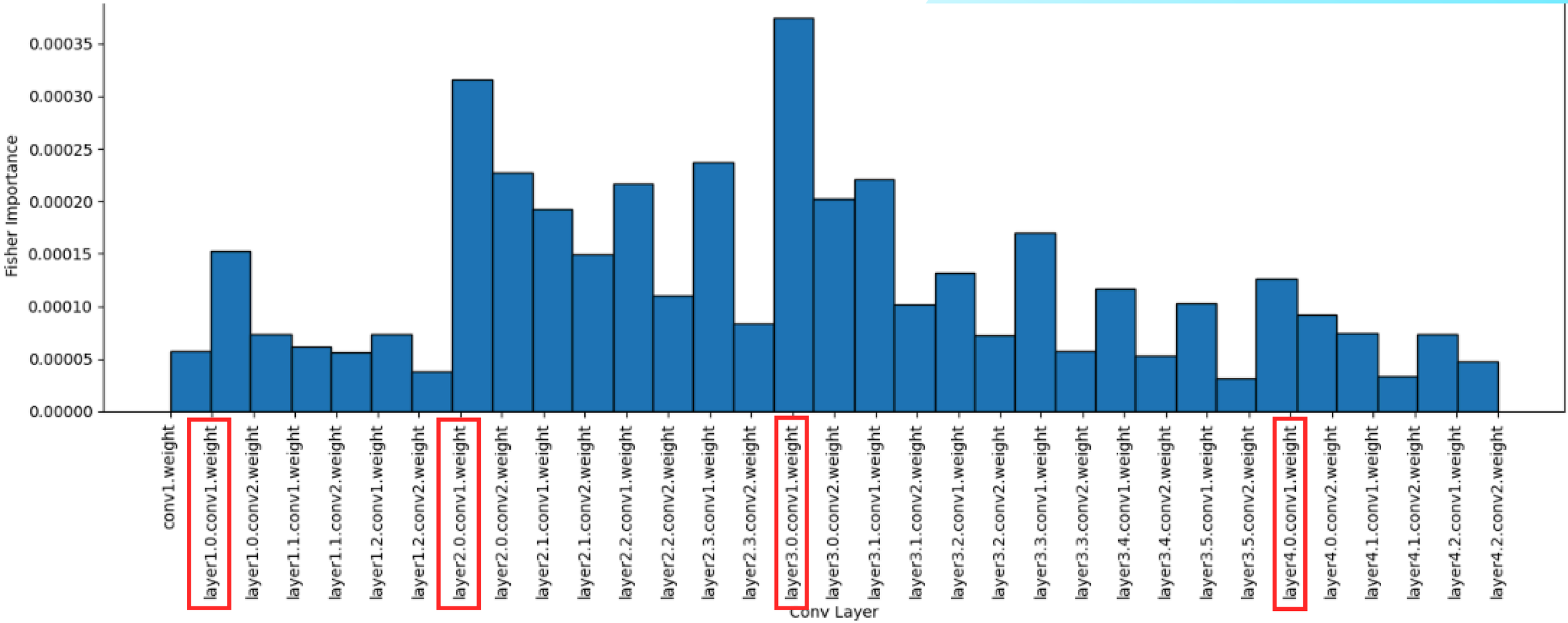
- Architecture: ResNet34 (pretrained), buffer 200 samples/class
- 4 tasks, 2 classes per task, 100 epochs per task
- Average accuracy: 58.89%





Visualization for CIFAR-10

- Fisher information importance by conv layers after the last task
- The conv layer at the beginning of each “block” has the highest importance with respect to that block

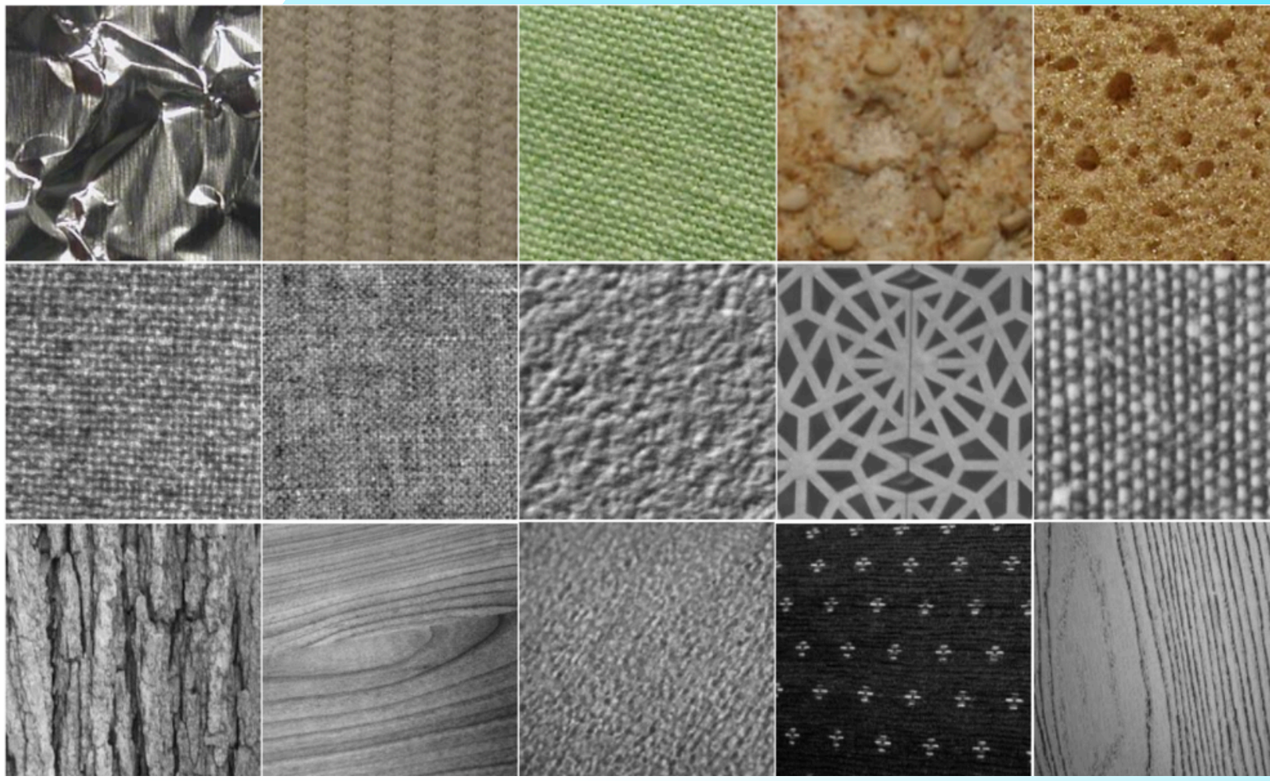


Experiment on other defect dataset

Texture dataset

A dataset of surface texture (8674 images), which contains 64 classes from 3 public datasets. The images are uniformly resized into 331 x 331.

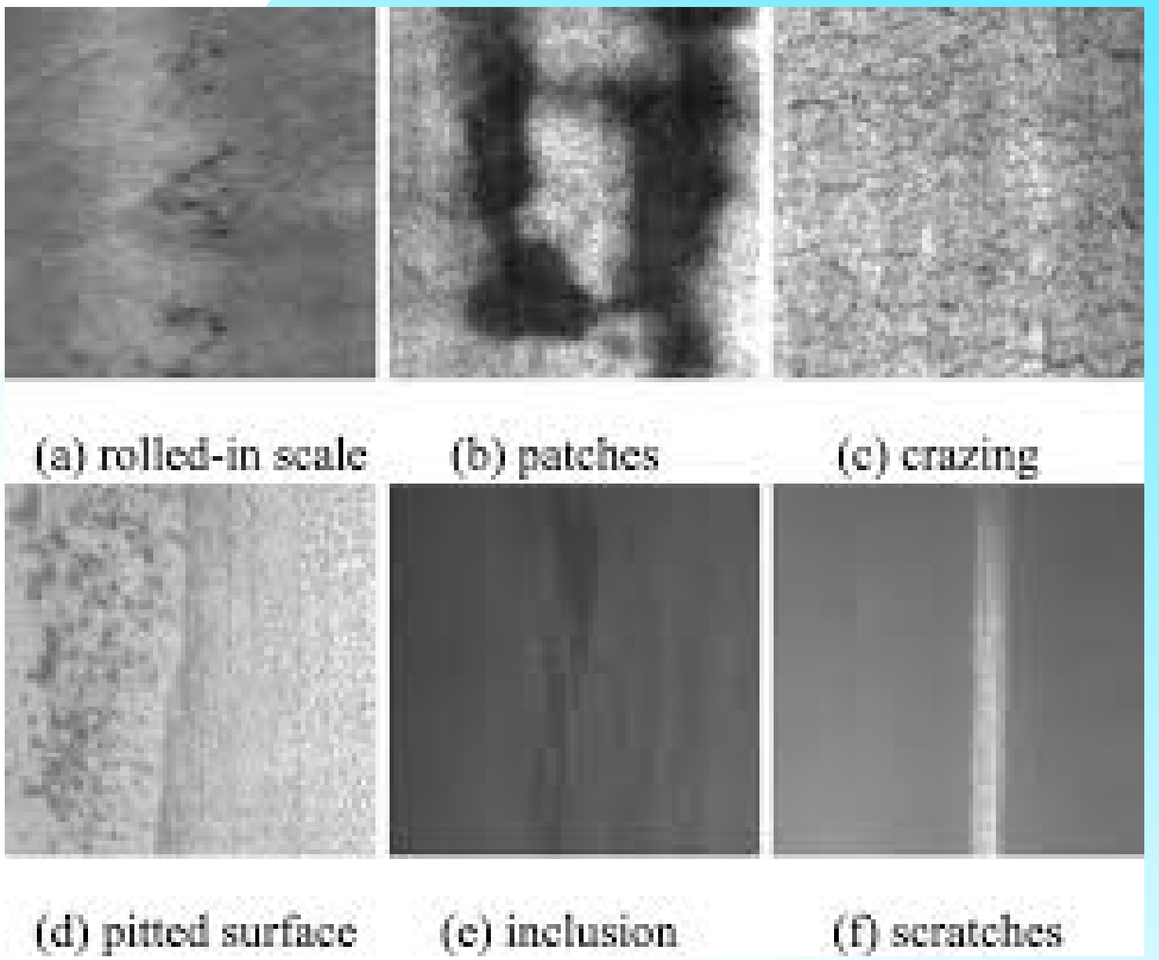
These images are real-world surface texture from wood, blanket, cloth, leather, etc, it can be used as a pretrain dataset to improve the performance of the CNN models for surface defect inspection.



Experiment on other defect dataset

NEU dataset

- Developed by: Northeastern University (NEU), China
- Purpose: Research on automated surface defect detection in industrial materials
- Application domain: Hot-rolled steel strip surface inspection
- Number of images: 1,800
- Image format: Grayscale
- Image resolution: 200 × 200 pixels
- Number of defect classes: 6



Experiment on other defect dataset

NCAT12-DET dataset

- NCAT12-DET is a comprehensive surface defect dataset that was collected on cars.
- It comprises 7,200 high-resolution images across 12 distinct defect categories
- Classes: burn, clear glass, corrosion, cracks, cutouts, defaced paint, dents, hairline glass, inclusion, paint-peel, spidery glass, and tempered glass
- Dataset: https://github.com/Brym-Gyimah/NCAT12-DET

Experiment on other defect datasets

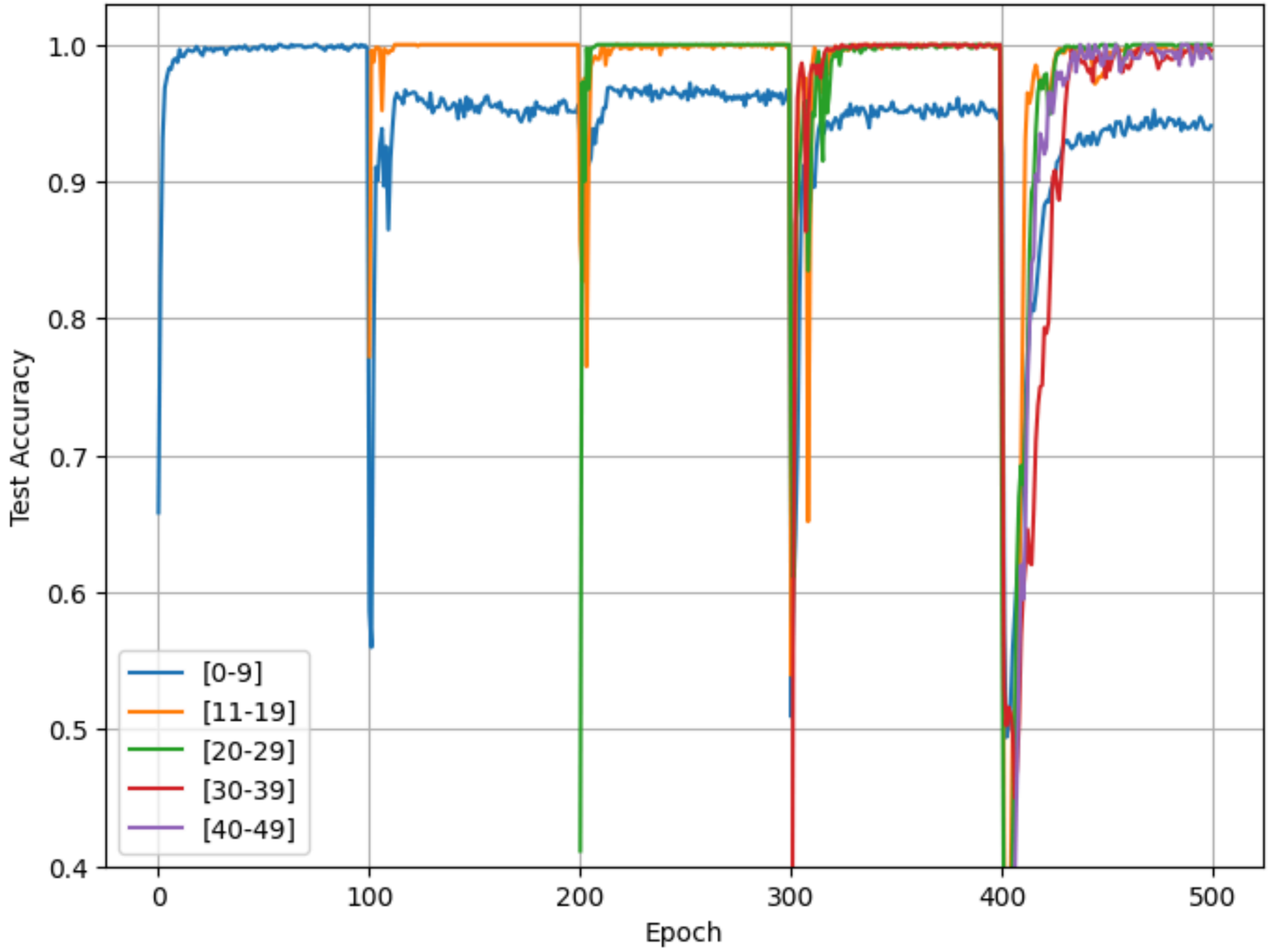
Results - Texture dataset

Settings:

- 5 tasks, 10 classes each
- Use last 4 classes as OOD
- 100 epochs/task, lr 0.001
- $\lambda_{ewc} = \lambda_{ood} = 1.0, \mathrm{FPR} = 0.2$

Accuracy by each task:

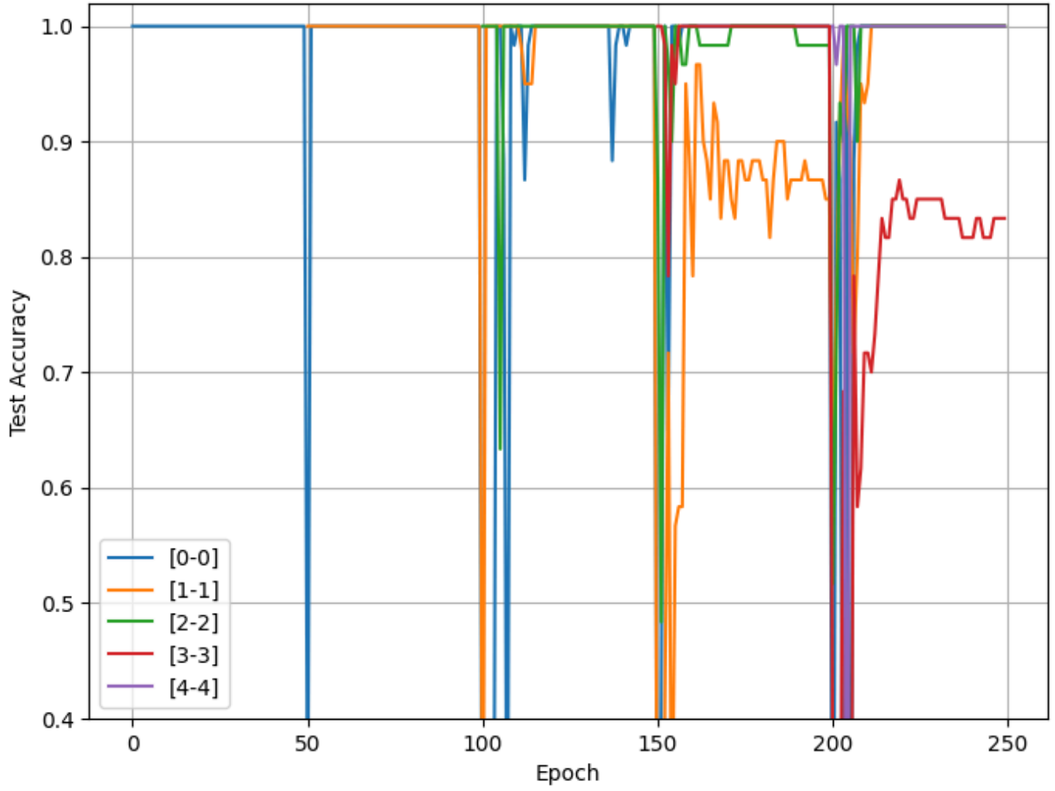
- Task 0: 94.08
- Task 1: 99.57
- Task 2: 100.0
- Task 3: 99.59
- Task 4: 99.0



Experiment on other defect datasets

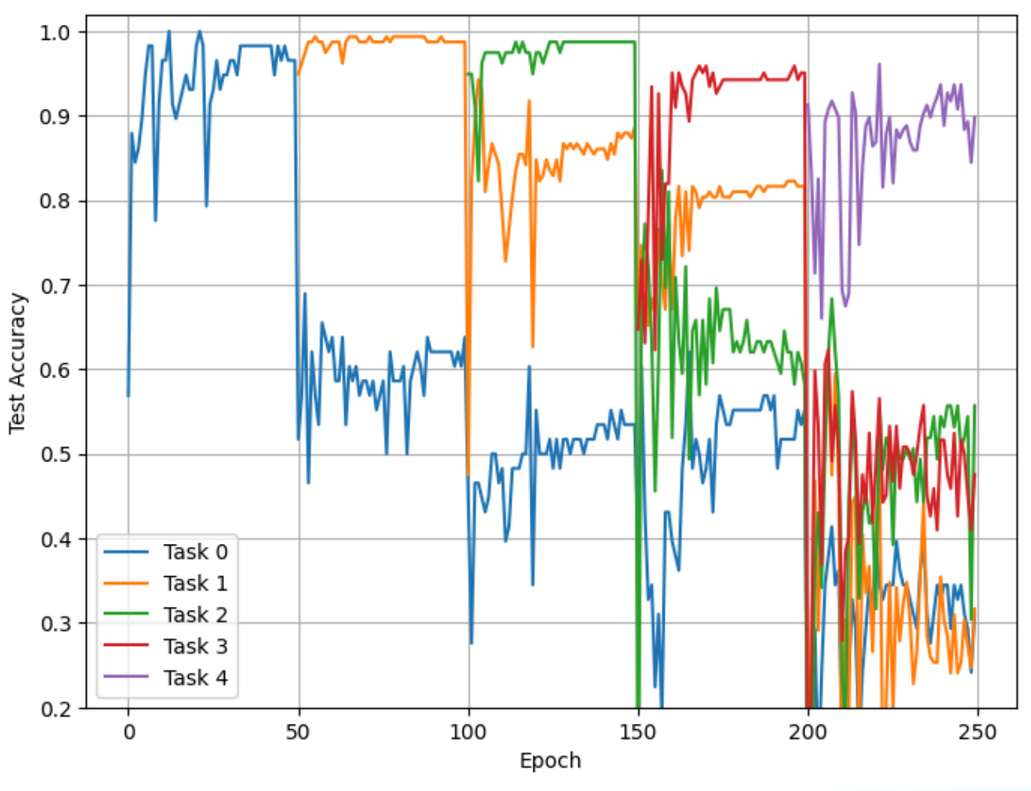
Results - NEU and NCAT-12 dataset

- 5 tasks, 1 classes each
- Use last class as OOD
- 50 epochs/task, lr 0.001



NEU dataset

- 5 tasks, 2 classes each
- Use last 2 classes as OOD
- 50 epochs/task, lr 0.001



NCAT-12 dataset

Planning

- Enhance current buffer strategy (random) to store exemplars that nearest to class mean (iCARL-style buffer).
- Integrate contrastive learning into the current framework.
- Test the model on metrics other than average accuracy (like Forgetting measure and Backward Transfer) - models can have the same accuracy, but some have lower forgetting than the others
- Code: https://www.kaggle.com/code/nixpaulo/adaptive-defect-classification