Continual Learning Using Out-of-Distribution Detection

Gyuhak Kim

Overview

- Motivations, Definitions, problem setups, related works
- A theoretical understanding
- Proposed methods
  - Parameter-isolation
  - Replay-based method

Motivation - Continual Learning

- Learning new tasks without forgetting the previous knowledge
  - with limited resources
- This is challenging due to catastrophic forgetting

Definition - CIL

Class incremental learning (CIL). CIL learns a sequence of tasks, $1, 2, ..., T$. Each task k has a training dataset $\mathcal{D}_k = \{(x_k^i, y_k^i)_{i=1}^{n_k}\}$, where n_k is the number of data samples in task k, and $x_k^i \in \mathbf{X}$ is an input sample and $y_k^i \in \mathbf{Y}_k$ (the set of all classes of task k) is its class label. All $\mathbf{Y}_k$'s are disjoint ($\mathbf{Y}_k \cap \mathbf{Y}_{k'} = \emptyset, \forall k \neq k'$) and $\bigcup_{k=1}^{T} \mathbf{Y}_k = \mathbf{Y}$. The goal of CIL is to construct a single predictive function or classifier $f : \mathbf{X} \rightarrow \mathbf{Y}$ that can identify the class label y of each given test instance x.

- Given x, what is its class?

Definition - TIL

Task incremental learning (TIL). TIL learns a sequence of tasks, $1, 2, ..., T$. Each task k has a training dataset $\mathcal{D}_k = \{((x_k^i, k), y_k^i)_{i=1}^{n_k}\}$, where n_k is the number of data samples in task $k \in \mathbf{T} = \{1, 2, ..., T\}$, and $x_k^i \in \mathbf{X}$ is an input sample and $y_k^i \in \mathbf{Y}_k \subset \mathbf{Y}$ is its class label. The goal of TIL is to construct a predictor $f : \mathbf{X} \times \mathbf{T} \rightarrow \mathbf{Y}$ to identify the class label $y \in \mathbf{Y}_k$ for (x, k) (the given test instance x from task k).

- Given that x is from task t, what is its class?
- We are interested in CIL

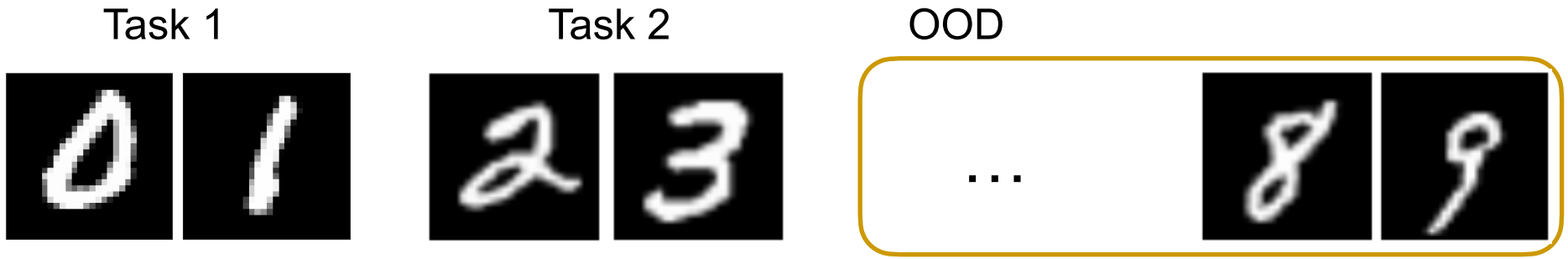
Definition - Out-of-Distribution Detection

- Out-of-Distribution (OOD) detection
  - Assign a class if an instance belongs to one of the classes used in the training data (IND)
  - Reject if an instance does not belong to any of the n IND training classes

Definition - Out-of-Distribution Detection

- For task 2, OOD classes are

- For task 1 and 2, OOD classes are



Definition - Out-of-Distribution Detection

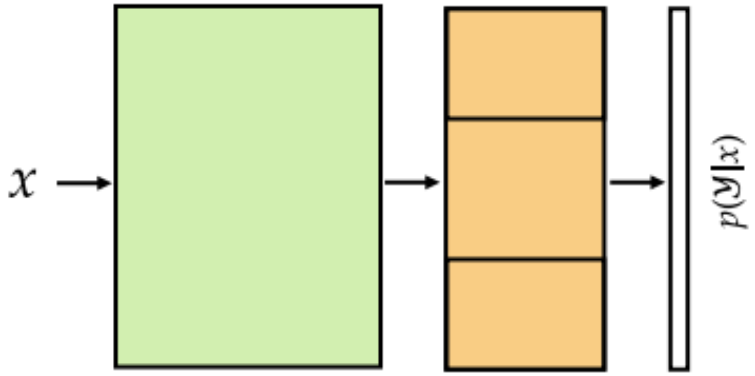
- The definition of OOD score depends on application
  - The popular definition is the maximum probability over IND classes

$$\max_{y} p(y|x)$$

Related Work

- Regularization methods
- Replay-based methods
- Parameter-isolation methods

Related Work - Regularization Methods



- Used for CIL (but any CIL can be used for TIL)
- Exemplar-free
- For task k, minimize

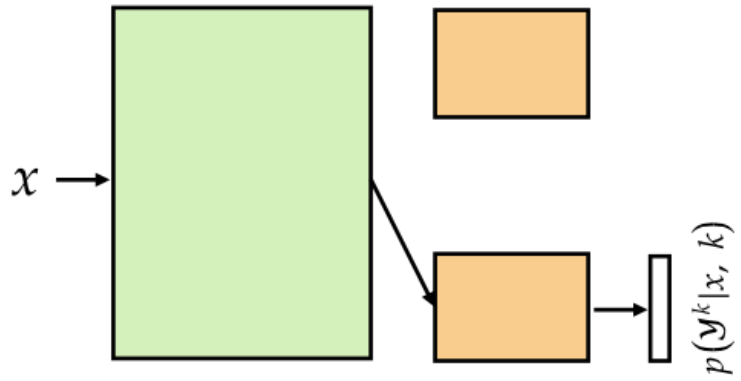
$$\mathcal{L} = -\frac{1}{|D^k|} \sum_{(x,y)\in D^k} \log p(y|x) + \mathcal{R}$$

Related Work - Replay-Based Methods

- Used for CIL (but any CIL can be used for TIL)
- Memory buffer $\mathcal{M}$
- For task k, minimize

$$\mathcal{L} = -\left(\frac{1}{|D^k|} \sum_{(x,y)\in D^k} \log p(y|x) + \frac{1}{|\mathcal{M}|} \sum_{(x,y)\in \mathcal{M}} \log p(y|x) \right) + \mathcal{R}$$

Related Work - Parameter-Isolation



- Used for TIL
- Use task-specific parameters
- For task k, minimize

$$\mathcal{L} = -\frac{1}{|D^k|} \sum_{(x,y)\in D^k} \log p(y|x,k) + \mathcal{R}$$

A Theoretical Understanding

- CIL upper bound
- Relationship between CIL, TIL, and OOD detection
- Necessary and sufficient conditions for CIL
- Empirical validation

CIL Decomposition

- CIL problem can be decomposed into two subproblems: within-task prediction (WP) and task-id prediction (TP)

$$\mathbf{P}(x \in \mathbf{X}_{k_0,j_0}|D) = \sum_{k=1,\ldots,n} \mathbf{P}(x \in \mathbf{X}_{k,j_0}|x \in \mathbf{X}_k, D)\mathbf{P}(x \in \mathbf{X}_k|D)$$

$$= \mathbf{P}(x \in \mathbf{X}_{k_0,j_0}|\underbrace{x \in \mathbf{X}_{k_0}}, D)\mathbf{P}(\underbrace{x \in \mathbf{X}_{k_0}}|D)$$

WP (i.e., TIL) TP

Upper Bound of CIL Loss

$$H_{WP}(x) = H(\tilde{y}, \{\mathbf{P}(x \in \mathbf{X}_{k_0,j} | x \in \mathbf{X}_{k_0}, D)\}_j),$$
$$H_{CIL}(x) = H(y, \{\mathbf{P}(x \in \mathbf{X}_{k,j} | D)\}_{k,j}),$$
$$H_{TP}(x) = H(\bar{y}, \{\mathbf{P}(x \in \mathbf{X}_k | D)\}_k)$$

- The loss of CIL is bounded by that of WP and TP

Theorem 1. *If* $H_{TP}(x) \leq \delta$ *and* $H_{WP}(x) \leq \epsilon$, *we have* $H_{CIL}(x) \leq \epsilon + \delta$.

- CIL improves with WP or TP

Upper Bound of CIL Loss

$$H_{OOD,k}(x) = \begin{cases} H(1, \mathbf{P}'_k(x \in \mathbf{X}_k|D)) = -\log \mathbf{P}'_k(x \in \mathbf{X}_k|D), \ x \in \mathbf{X}_k, \\ H(0, \mathbf{P}'_k(x \in \mathbf{X}_k|D)) = -\log \mathbf{P}'_k(x \notin \mathbf{X}_k|D), \ x \notin \mathbf{X}_k. \end{cases}$$

- TP and out-of-distribution (OOD) detection bound each other

Theorem 2. *i) If* $H_{TP}(x) \leq \delta$, *let* $\mathbf{P}'_k(x \in \mathbf{X}_k|D) = \mathbf{P}(x \in \mathbf{X}_k|D)$, *then* $H_{OOD,k}(x) \leq \delta, \forall\, k = 1, \ldots, T$. *ii) If* $H_{OOD,k}(x) \leq \delta_k, k = 1, \ldots, T$, *let* $\mathbf{P}(x \in \mathbf{X}_k|D) = \frac{\mathbf{P}'_k(x \in \mathbf{X}_k|D)}{\sum_k \mathbf{P}'_k(x \in \mathbf{X}_k|D)}$, *then* $H_{TP}(x) \leq (\sum_k \mathbf{1}_{x \in \mathbf{X}_k} e^{\delta_k})(\sum_k 1 - e^{-\delta_k})$, *where* $\mathbf{1}_{x \in \mathbf{X}_k}$ *is an indicator function.*

Upper Bound of CIL Loss

- The loss of CIL is bounded by that of WP and OOD

Theorem 3. *If* $H_{OOD,k}(x) \leq \delta_k,\ k = 1, \ldots, T$ *and* $H_{WP}(x) \leq \epsilon$, *we have*

$$H_{CIL}(x) \leq \epsilon + (\sum_k \mathbf{1}_{x \in \mathbf{X}_k} e^{\delta_k})(\sum_k 1 - e^{-\delta_k}),$$

where $\mathbf{1}_{x \in \mathbf{X}_k}$ *is an indicator function.*

Necessary Condition for CIL

- Previously, we showed that good performances of WP and TP or (OOD) are *sufficient* to guarantee a good CIL

- Good performances of WP and TP (or OOD) are *necessary* for a good CIL

Theorem 4. *If* $H_{CIL}(x) \leq \eta$, *then there exist i) a WP, s.t.* $H_{WP}(x) \leq \eta$, *ii) a TP, s.t.* $H_{TP}(x) \leq \eta$, *and iii) an OOD detector for each task, s.t.* $H_{OOD,k} \leq \eta$, $k = 1, \ldots, T$.

Empirical Validation

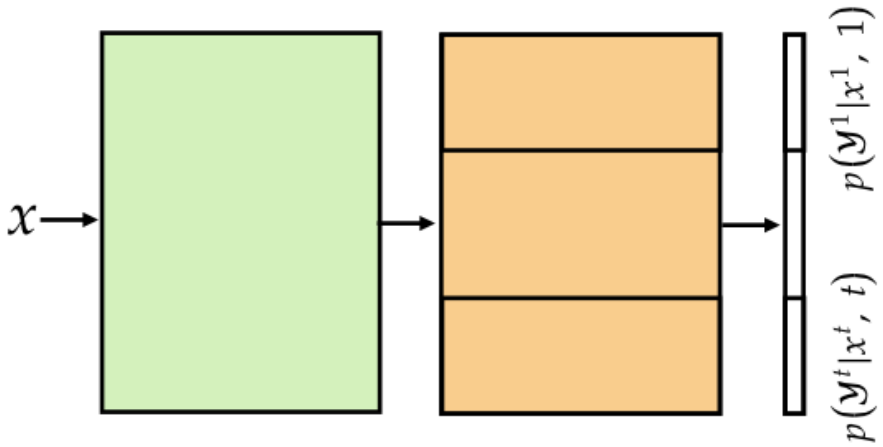
- Evaluation metrics:
  - Average classification accuracy: accuracy over all the learned classes
  - Average Area Under the ROC Curve (AUC) over all the tasks

$$AUC = \sum_k AUC_k / n$$

- CL datasets/splits: MNIST-5T, CIFAR10-5T, CIFAR100-10T, 20T, Tiny-ImageNet-5T, 10T

Empirical Validation

- How to convert the baselines



- We want to show that OOD improves CIL

Empirical Validation

- We want to show that OOD improves CIL:
  - Post-processing CIL models with OOD detection (ODIN)
    - Temperature scaling

$$s(x;\tau_k)_j = e^{f(x)_{kj}/\tau_k} / \sum_j e^{f(x)_{kj}/\tau_k}$$

    - Positive noise

$$\tilde{x} = x - \epsilon_k \text{sign}(-\nabla_x \log s(x;\tau_k)_{\hat{y}})$$

Empirical Validation

- CIL accuracy increases and decreases by the OOD detection performance (AUC)

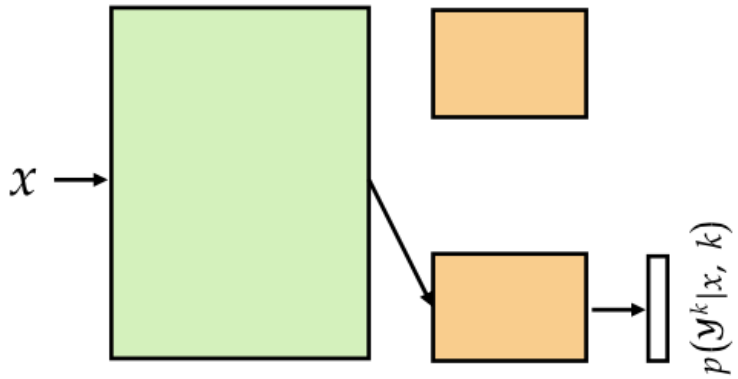
Method	OOD	AUC	CIL
OWM	Original	71.31	28.91
	ODIN	70.06	28.88
MUC	Original	72.69	30.42
	ODIN	72.53	29.79
PASS	Original	69.89	33.00
	ODIN	69.60	31.00
LwF	Original	88.30	45.26
	ODIN	87.11	51.82
BiC	Original	87.89	52.92
	ODIN	86.73	48.65
DER++	Original	85.99	53.71
	ODIN	88.21	55.29
HAT	Original	77.72	41.06
	ODIN	77.80	41.21
HyperNet	Original	71.82	30.23
	ODIN	72.32	30.83
Sup	Original	79.16	44.58
	ODIN	80.58	46.74

Proposed Methods (1) - Overview

- Motivation
- Proposed methods
  - Hard Attention to the Task
  - Supermasks in Superposition
  - Contrasting Shifted Instances
- Experiment

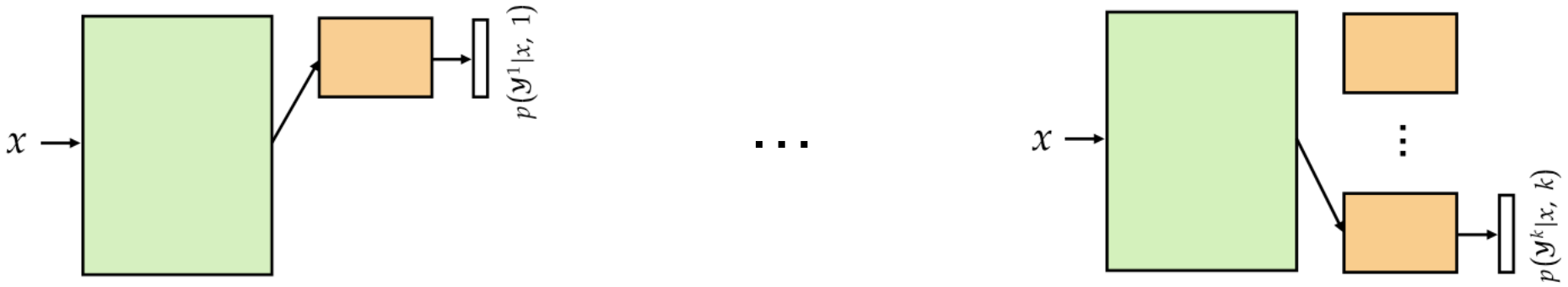
Motivation

- CIL improves with WP (i.e., TIL) or TP (i.e., OOD for each task)
- Strong TIL methods + strong OOD methods
  - Parameter-isolation methods



Motivation

- Parameter-isolation methods
  - TIL is about learning a function (for a task) without forgetting



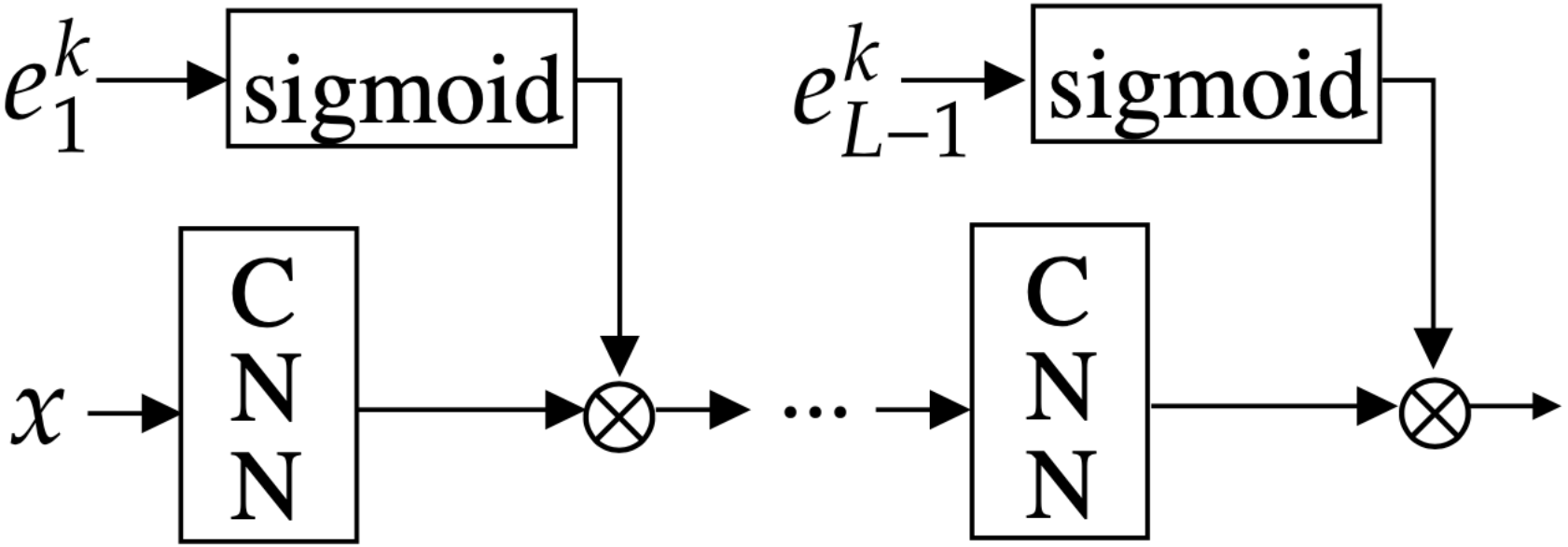
Motivation

- Parameter-isolation methods
  - TIL is about learning a function (for a task) without forgetting

- OOD is about learning an OOD detection (for a task)

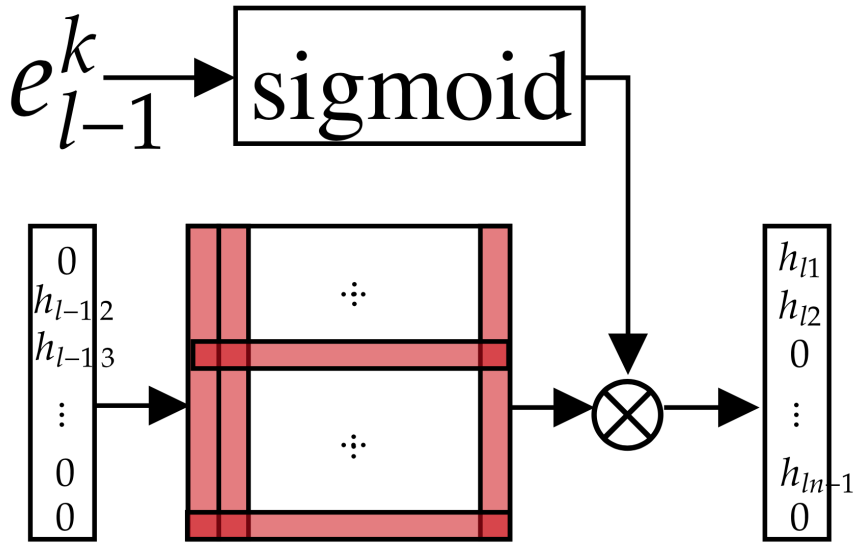
Parameter-Isolation: HAT (Serra et al., 2018)

- Hard Attention to the Task (HAT)
  - Inside feature extractor:



Parameter-Isolation: HAT (Serra et al., 2018)

- Hard Attention to the Task (HAT)
  - Inside feature extractor:



  - Prevent parameter change

Parameter-Isolation: SupSup (Wortsman et al., 2020)

- Supermasks in Superposition (SupSup)
  - Finding “supermask” for each task
  - Supermask - a subnetwork of a randomly initialized neural network that achieves high accuracy without training (Lottery Ticket Hypothesis)

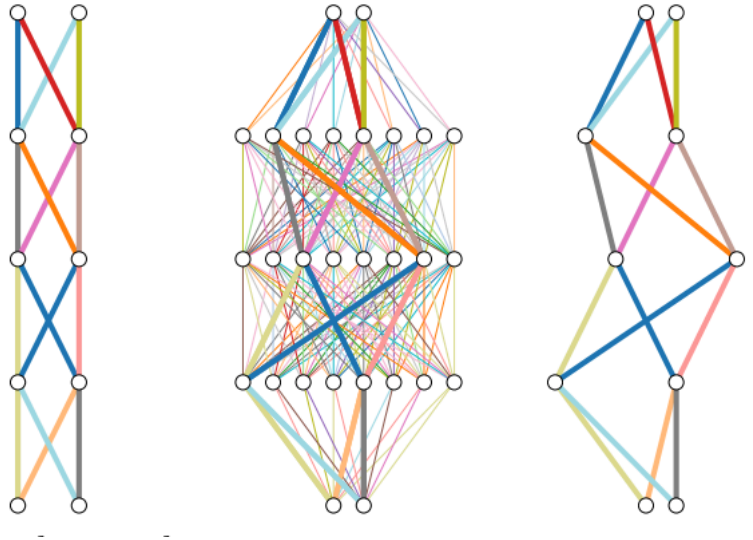


Image from Figure 1 (Ramanujan et al. 2020)

Parameter-Isolation: SupSup (Wortsman et al., 2020)

- Supermasks in Superposition (SupSup)
  - Proposed algorithm: Edge Popup (Ramanujan, et al., 2020)

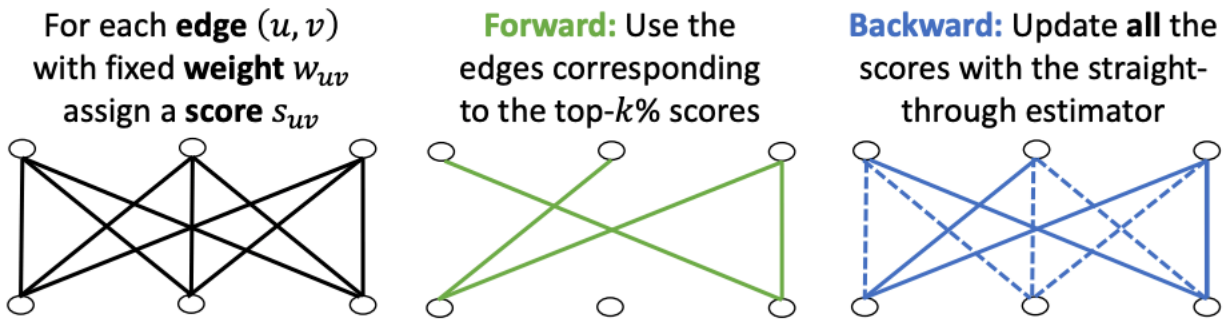


Image from Figure 2 (Ramanujan et al. 2020)

OOD Detection - CSI (Tack et al., 2020)

- Constrasting Shifted Instances (CSI)
  - Use supervised contrastive loss

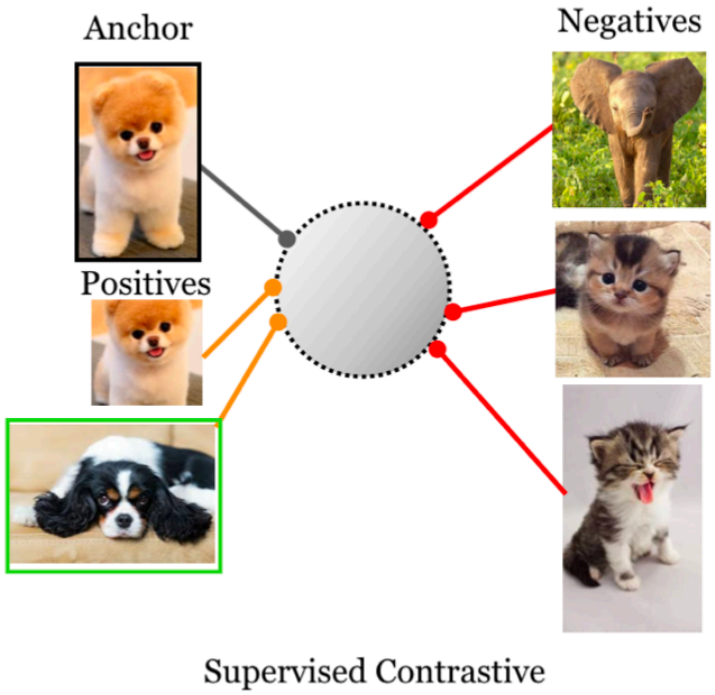


Image from Figure 2 (Khosla et al. 2020)

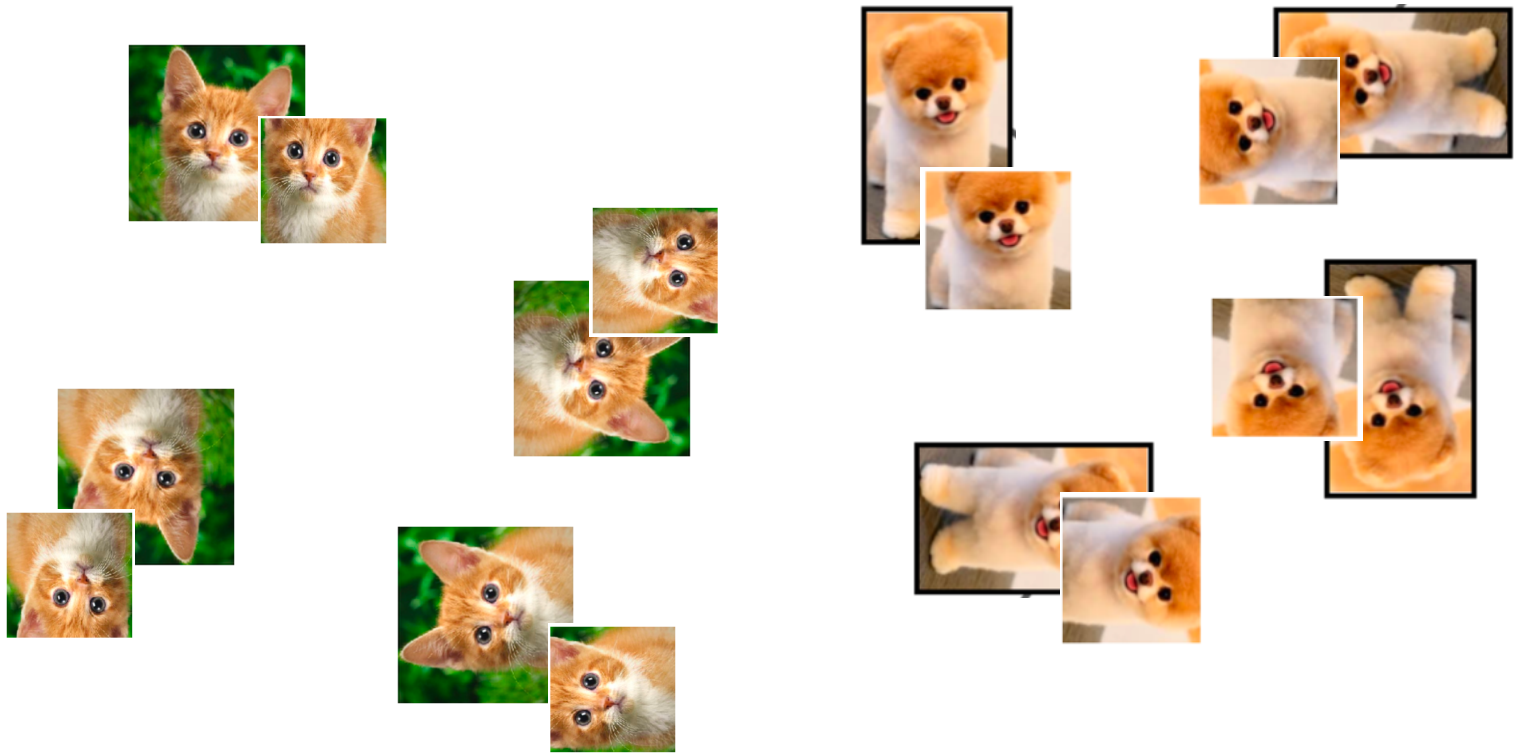
OOD Detection - CSI (Tack et al., 2020)

- Constrasting Shifted Instances (CSI)
  - A lot of augmentations

Image from Tack et al. 2020

OOD Detection - CSI (Tack et al., 2020)

- Constrasting Shifted Instances (CSI)
  - Learning classes and rotations



OOD Detection - CSI (Tack et al., 2020)

- Constrasting Shifted Instances (CSI)
  - We are interested in ‘class’ rather than ‘class_90’, ‘class_180’, …
  - Ensemble output for class j of task k over all degrees

$$f(h(x,k))_{j_k} = \frac{1}{4}\sum_{\text{deg}} f(h(x_{\text{deg}},k))_{j_k,\text{deg}}$$

Proposed Method - TIL + OOD

- Now, train each task for OOD detection (CSI)
  - Protect each network using TIL methods (HAT or Sup)
  - Each network produces good TIL performance and good OOD performance
    - This is the desired property of CIL

Experiment

- HAT+CSI and Sup+CSI
  - Better OOD method (CSI) results in better CIL

CL	OOD	C10-5T		C100-10T		C100-20T		T-5T		T-10T	
		AUC	CIL	AUC	CIL	AUC	CIL	AUC	CIL	AUC	CIL
HAT	ODIN	82.5	62.6	77.8	41.2	75.4	25.8	72.3	38.6	71.8	30.0
	CSI	91.2	87.8	84.5	63.3	86.5	54.6	76.5	45.7	78.5	47.1
Sup	ODIN	82.4	62.6	80.6	46.7	81.6	36.4	74.0	41.1	74.6	36.5
	CSI	91.6	86.0	86.8	65.1	88.3	60.2	77.1	48.9	79.4	45.7

Experiment

- Comparison (TIL)

Method	M-5T	C10-5T	C100-10T	C100-20T	T-5T	T-10T
DER++	99.7±0.08	92.0±0.54	84.0±9.43	86.6±9.44	57.4±1.31	60.0±0.74
HAT	99.9±0.02	96.7±0.18	84.0±0.23	85.0±0.98	61.2±0.72	63.8±0.41
Sup	99.6±0.01	96.6±0.21	87.9±0.27	91.6±0.15	64.3±0.24	68.4±0.22
HAT+CSI	99.9±0.00	98.7±0.06	92.0±0.37	94.3±0.06	68.4±0.16	72.4±0.21
Sup+CSI	99.0±0.08	98.7±0.07	93.0±0.13	95.3±0.20	65.9±0.25	74.1±0.28

Experiment

- Comparison (CIL)

Method	M-5T	C10-5T	C100-10T	C100-20T	T-5T	T-10T
OWM	95.8±0.13	51.8±0.05	28.9±0.60	24.1±0.26	10.0±0.55	8.6±0.42
MUC	74.9±0.46	52.9±1.03	30.4±1.18	14.2±0.30	33.6±0.19	17.4±0.17
PASS†	76.6±1.67	47.3±0.98	33.0±0.58	25.0±0.69	28.4±0.51	19.1±0.46
LwF	85.5±3.11	54.7±1.18	45.3±0.75	44.3±0.46	32.2±0.50	24.3±0.26
iCaRL*	96.0±0.43	63.4±1.11	51.4±0.99	47.8±0.48	37.0±0.41	28.3±0.18
Mnemonics†*	96.3±0.36	64.1±1.47	51.0±0.34	47.6±0.74	37.1±0.46	28.5±0.72
BiC	94.1±0.65	61.4±1.74	52.9±0.64	48.9±0.54	41.7±0.74	33.8±0.40
DER++	95.3±0.69	66.0±1.20	53.7±1.30	46.6±1.44	35.8±0.77	30.5±0.47
Co^{2}L		65.6				
CCG	97.3	70.1				
HAT	81.9±3.74	62.7±1.45	41.1±0.93	25.6±0.51	38.5±1.85	29.8±0.65
HyperNet	56.6±4.85	53.4±2.19	30.2±1.54	18.7±1.10	7.9±0.69	5.3±0.50
Sup	70.1±1.51	62.4±1.45	44.6±0.44	34.7±0.30	41.8±1.50	36.5±0.36
PR-Ent	74.1	61.9	45.2			
HAT+CSI	94.4±0.26	87.8±0.71	63.3±1.00	54.6±0.92	45.7±0.26	47.1±0.18
Sup+CSI	80.7±2.71	86.0±0.41	65.1±0.39	60.2±0.51	48.9±0.25	45.7±0.76
HAT+CSI+c	96.9±0.30	88.0±0.48	65.2±0.71	58.0±0.45	51.7±0.37	47.6±0.32
Sup+CSI+c	81.0±2.30	87.3±0.37	65.2±0.37	60.5±0.64	49.2±0.28	46.2±0.53

Proposed Method (2) - Overview

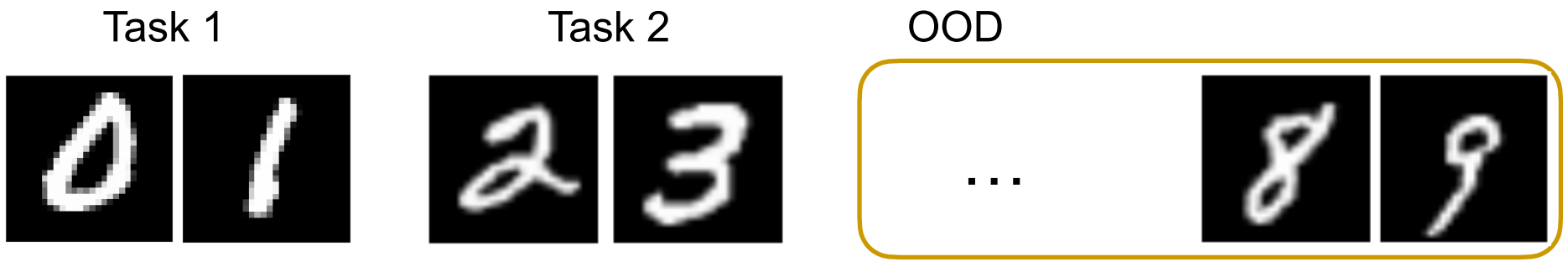
- Motivation
- Proposed methods
  - Replay-based OOD detection model
  - Updating previous task model
  - Improving the performance by a distance-based technique
- Experiment

Motivation

- Leverage strong pre-trained models
  - CSI cannot be used with pre-trained models
- For longer tasks (i.e., small number of classes per task), the performance was low
  - e.g., 65% in CIFAR100-10T vs 60% in CIFAR100-20T
- OOD detection in CL setting

Motivation

- OOD detection in CL setting
  - After learning 2nd task, IND are classes of task 1 and 2, and OOD are any classes not from task 1 and 2



  - Make the system fully autonomous

Motivation

- Replay-based methods are highly effective
  - Propose a replay-based method based on the design framework
- What is replay-based method?
  - Memory buffer $\mathcal{M}$
  - For task k, minimise

$$\mathcal{L} = -\left(\frac{1}{|D^k|} \sum_{(x,y)\in D^k} \log p(y|x) + \frac{1}{|\mathcal{M}|} \sum_{(x,y)\in \mathcal{M}} \log p(y|x) \right) + \mathcal{R}$$

Training OOD with Replay Buffer

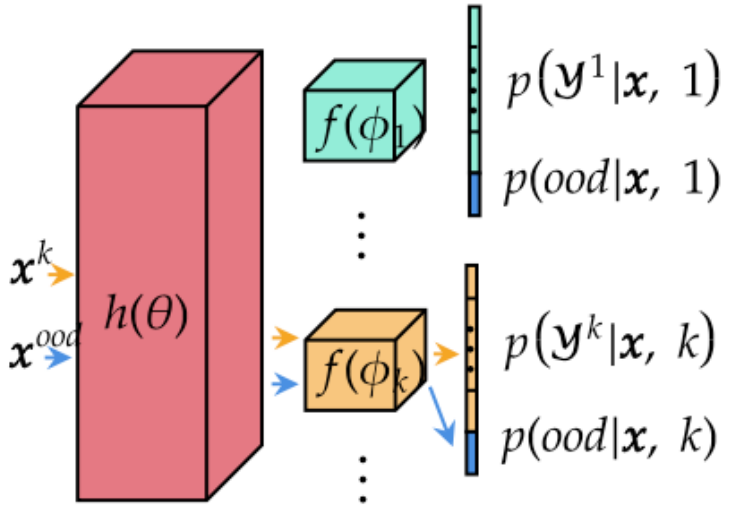
- Train each task network to predict IND classes and OOD classes
  - Train each task network to predict the IND classes and OOD classes (i.e., any classes that do not belong to the task)
  - Samples of previous tasks are saved in a memory buffer $\mathcal{M}$
  - When training task k, minimize

$$\mathcal{L}_{ood}(\theta, \phi_t) = -\frac{1}{M+N}\left(\sum_{(x,y)\in\mathcal{M}} \log p(ood|x,k) + \sum_{(x,y)\in\mathcal{D}^k} \log p(y|x,k)\right)$$

- Use the parameter-isolation (HAT) to prevent forgetting

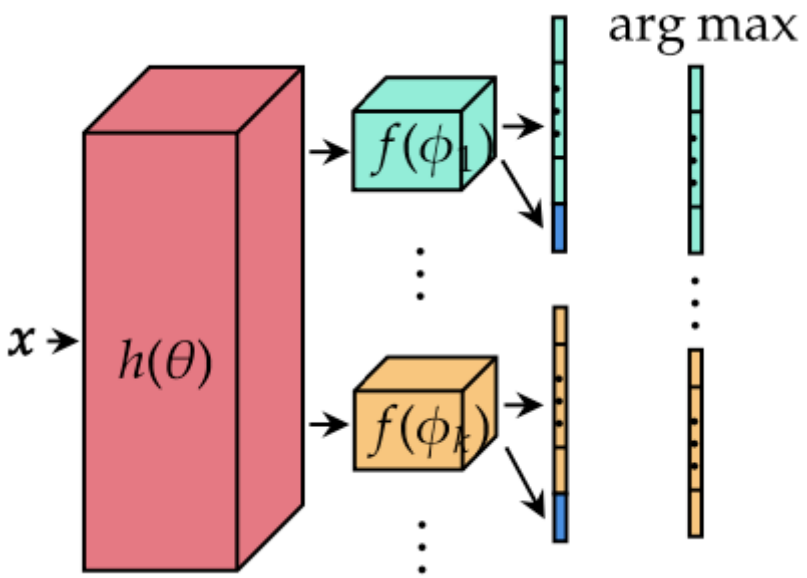
Training OOD with Replay Buffer

- Previously, each task network is trained to predict classes (IND) of a task
  - Train each task network to predict the IND classes and OOD classes



Training OOD with Replay Buffer

- At inference, we do not consider the OOD class, but consider only the IND classes from each task.



Updating Previous Task Models

- Later task models are better OOD detection than the earlier ones as the later ones are trained with more diverse OOD classes
  - Task 1 is only trained with IND classes
  - Task 2 is trained with IND classes and classes from task 1 (OOD)
  - Task 3 is trained with IND classes and classes from task 1 and 2 (OOD)
  - …
- Improve the earlier task models

Updating Previous Task Models

- Update each task model to incorporate new OOD classes into its classifier

- For each task j, minimize

$$\mathcal{L}(\phi_j) = -\frac{1}{2M}\left(\sum_{(x,y)\in\tilde{\mathcal{M}}} \log p(ood|x,j) + \sum_{(x,y)\in\tilde{\mathcal{D}}^k} \log p(y|x,j)\right)$$

Improving the Performance - Distance-Based Coef

- If a test instance is close to a class (in feature space), it is more likely to belong to the task
- Use Mahalanobis distance to measure the distance
- For task t, the mean of a class j and the variance are

$$\mu_j^t = \sum_{x \in \mathcal{D}_j^t} h(x, j)/|\mathcal{D}_j^t| \qquad S^t = \sum_{j \in \mathcal{Y}^t} S_j^t/|\mathcal{Y}^t|$$

Improving the Performance - Distance-Based Coef

- Mahalanobis distance (MD) of an instance x to the distribution $N(\mu_j^t, S^t)$ is $\mathrm{MD}(x; \mu_j^t, S^t)$
- The smaller the MD, the closer the instance to the distribution
- This measures the OOD-ness of a sample in the feature space to task k

Improving the Performance - Distance-Based Coef

- Define the coefficient as

$$s^t(x) = \max\left[1/\text{MD}(x; \mu^t_{y_1}, S^t), \cdots, 1/\text{MD}(x; \mu^t_{y_{|\mathcal{Y}^t|}}, S^t)\right]$$

- s is small if x is OOD to task t, and s is large if it is IND to task t

- At prediction,

$$y = \arg\max \bigoplus_{1 \le k \le t} p(\mathcal{Y}^k|x, k) s^k(x)$$

Experiment

- Same CL experiment as before (C10-5T, C100-10T, 20T, T-5T, 10T)
- Pre-trained network
  - Pre-train without overlapping classes. 618 classes of ImageNet after removing 382 classes similar/identical to CIFAR and Tiny-ImageNet
  - Fixed feature extractor, trainable adapter modules
  - Same architecture for the baselines for fairness

Experiment

- Evaluation metrics:
  - Average classification accuracy
  - AUC in CL setting (at second last task)
    - e.g., For CL problem with 5 tasks, evaluate AUC after learning the 4th task.
    - Classes from task 1 to 4 are IND and classes from 5th task are OOD

Experiment

- Better performance. CIFAR100-10T and 20T are about the same

Method	C10-5T	C100-10T	C100-20T	T-5T	T-10T	Avg.
OWM	41.69±6.34	21.39±3.18	16.98±4.44	24.55±2.48	17.52±3.45	24.43
PASS	86.21±1.10	68.90±0.94	66.77±1.18	61.03±0.38	58.34±0.42	68.25
iCaRL	87.55±0.99	68.90±0.47	69.15±0.99	53.13±1.04	51.88±2.36	66.12
A-GEM	56.33±7.77	25.21±4.00	21.99±4.01	30.53±3.99	21.90±5.52	31.20
EEIL	82.34±3.13	68.08±0.51	63.79±0.66	53.34±0.54	50.38±0.97	63.59
GD	**89.16**±0.53	64.36±0.57	60.10±0.74	53.01±0.97	42.48±2.53	61.82
DER++	84.63±2.91	69.73±0.99	70.03±1.46	55.84±2.21	54.20±3.28	66.89
HAL	84.38±2.70	67.17±1.50	67.37±1.45	52.80±2.37	55.25±3.60	65.39
HAT	83.30±1.54	62.34±0.93	56.72±0.44	57.91±0.72	53.12±0.94	62.68
MORE	**89.16**±0.96	**70.23**±2.27	**70.53**±1.09	**64.97**±1.28	**63.06**±1.26	**71.59**

Experiment

- Almost no performance reduction in small memory buffer

Method	C10-5T	C100-10T	C100-20T	T-5T	T-10T	Avg.
OWM	41.69±6.34	21.39±3.18	16.98±4.44	24.55±2.48	17.52±3.45	24.43
iCaRL	86.08±1.19	66.96±2.08	68.16±0.71	47.27±3.22	49.51±1.87	63.60
A-GEM	56.64±4.29	23.18±2.54	20.76±2.88	31.44±3.84	23.73±6.27	31.15
EEIL	77.44±3.04	62.95±0.68	57.86±0.74	48.36±1.38	44.59±1.72	58.24
GD	85.96±1.64	57.17±1.06	50.30±0.58	46.09±1.77	32.41±2.75	54.39
DER++	80.09±3.00	64.89±2.48	65.84±1.46	50.74±2.41	49.24±5.01	62.16
HAL	79.16±4.56	62.65±0.83	63.96±1.49	48.17±2.94	47.11±6.00	60.21
PASS	86.21±1.10	68.90±0.94	66.77±1.18	61.03±0.38	58.34±0.42	68.25
HAT	83.30±1.54	62.34±0.93	56.72±0.44	57.91±0.72	53.12±0.94	62.68
MORE	**88.13**±1.16	**71.69**±0.11	**71.29**±0.55	**64.17**±0.77	**61.90**±0.90	**71.44**

Experiment

- Better AUC in CL setting

Method	C10-5T	C100-10T	C100-20T	T-5T	T-10T	Avg.
OWM	58.26±17.38	50.87±2.86	55.43±10.25	58.20±2.51	56.17±4.26	55.79
iCaRL	78.54±9.59	72.10±2.66	69.79±5.75	66.05±1.73	66.62±1.77	70.19
A-GEM	63.71±15.18	52.18±2.60	54.78±13.40	58.97±2.52	56.33±4.14	57.19
EEIL	81.56±10.62	67.39±3.44	64.83±8.01	67.22±2.16	62.36±6.14	68.58
GD	**85.02**±9.88	64.22±2.47	61.95±9.02	68.35±2.97	58.79±3.10	67.67
DER++	79.25±4.74	70.36±1.81	69.74±2.02	68.67±3.83	67.81±0.23	70.93
HAL	77.97±9.76	69.55±0.83	71.58±3.54	67.58±3.71	67.27±1.86	70.79
PASS	77.69±4.01	71.80±2.41	66.62±5.78	71.61±1.10	68.51±4.49	71.24
HAT	83.89±4.10	71.26±1.93	65.52±3.43	75.08±1.07	72.02±1.35	73.55
MORE	80.83±8.82	**73.32**±2.80	**72.28**±4.81	**75.74**±2.66	**72.78**±1.08	**74.99**

Conclusion

- OOD detection is crucial for solving CL
- The system trained with OOD detection is naturally capable of OOD detection
  - The system now can detect novel classes, and learn the new classes continually.
  - Getting closer to the lifelong learning system

Future Work

- Using parameter-isolation methods is limited in certain learning scenario: revisiting samples
  - D1 = {dog, cat}, D2 = {dog, computer, car}

- Task splits are abstract. Can we find the best task split?
  - Can be useful for online CL where task boundaries are ambiguous

Reference

- Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*, 2018.
- Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Raste- gari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. In H. Larochelle, M. Ran- zato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *NeurIPS*, 2020.
- Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11893–11902, 2020.
- Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *NeurIPS*, 2020.
- Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In *NeurIPS*, 2020.

Thank you