Continuous Learning of Context-dependent Processing in Neural Networks

Nature Machine Intelligence

Presented by Yulai Cong
Feb 20, 2020

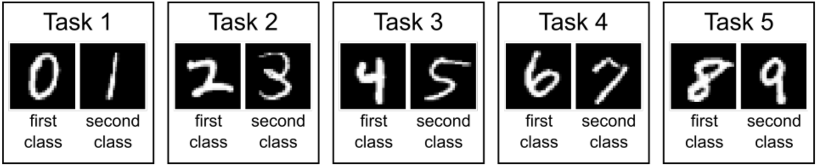
Table of Contents

The Lifelong/Continual Learning Problem

To continually learn to solve new tasks, while preserving the experiences learned on previous ones.
Challenge: Catastrophic forgetting of neural networks.

Basic Assumptions:

- Classification tasks.
- Tasks have clear and well-defined boundaries.



Categories of Lifelong/Continual Learning Problems

The availability of task identity at the test time [1]

Table 1: Overview of the three continual learning scenarios.

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided
Class-IL	Solve tasks so far *and* infer task-ID

[1] van de Ven, G. M., and Tolias, A. S. Three scenarios for continual learning. arXiv preprint arXiv:1904.07734, 2019.

Categories of Lifelong/Continual Learning Problems: Example 1/2

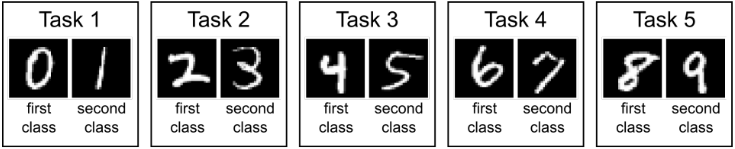


Figure 1: Schematic of split MNIST task protocol.

Table 2: Split MNIST according to each scenario.

Task-IL	With task given, is it the 1st or 2nd class? (e.g., 0 or 1)
Domain-IL	With task unknown, is it a 1st or 2nd class? (e.g., in [0, 2, 4, 6, 8] or in [1, 3, 5, 7, 9])
Class-IL	With task unknown, which digit is it? (i.e., choice from 0 to 9)

Categories of Lifelong/Continual Learning Problems: Example 2/2

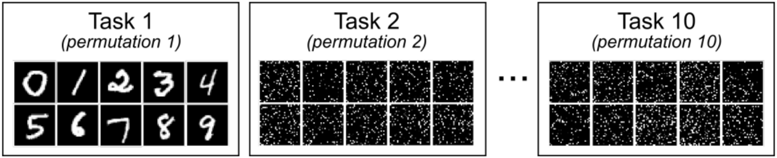
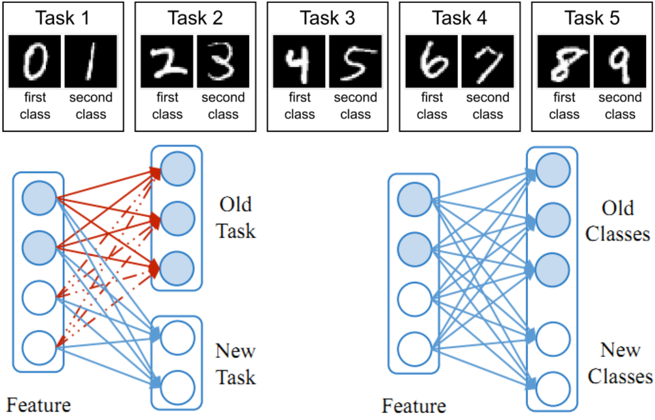


Figure 2: Schematic of permuted MNIST task protocol.

Table 3: Permuted MNIST according to each scenario.

Task-IL	Given permutation *X*, which digit?
Domain-IL	With permutation unknown, which digit?
Class-IL	Which digit *and* which permutation?

*Model Architectures



(b) Multi-head Output Layer (c) Single-head Output Layer

Table of Contents

A Rough Summary of Existing Methods

1. **Naive: Exact Replay & Coreset**
2. Model Architecture Manipulation
   - Context-Dependent Gating (XdG): Divide model capacity
   - Progress & Compress (P&C): Increase model capacity
3. Replay-Based: Distill previous ability/replay previous data
   - Learning without Forgetting (LwF)
   - Deep Generative Replay (DGR) & Distillation
   - Incremental Classifier and Representation Learning (iCaRL) + coreset
4. Regularization: Evaluate parameter importance
   - Elastic Weight Consolidation (EWC)
   - Online EWC
   - Synaptic Intelligence (SI)
   - *Variational Continual Learning (VCL) + coreset
5. Gradient Manipulation: the presented paper
   - Orthogonal Weights Modification (OWM)

Exact Replay & Coreset

Exact Replay: to remember all the data from all seen tasks.

- ✓ If applicable, the best performance is expected.
- × Usually non-applicable; privacy concerns or memory constraints.

A practical compromise: to remember the **coreset** of the data

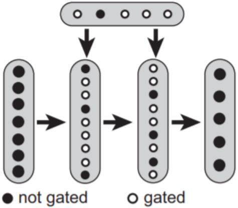
- A representative subset
- Uniformly selected samples
- The K centers of the data (the K-center algorithm)
- In practice, such a simple coreset idea works decently well.
- Not applicable if privacy concerns exist.

A Rough Summary of Existing Methods

1. Naive: Exact Replay & Coreset
2. **Model Architecture Manipulation**
   - Context-Dependent Gating (XdG): Divide model capacity
   - Progress & Compress (P&C): Increase model capacity
3. Replay-Based: Distill previous ability/replay previous data
   - Learning without Forgetting (LwF)
   - Deep Generative Replay (DGR) & Distillation
   - Incremental Classifier and Representation Learning (iCaRL) + coreset
4. Regularization: Evaluate parameter importance
   - Elastic Weight Consolidation (EWC)
   - Online EWC
   - Synaptic Intelligence (SI)
   - *Variational Continual Learning (VCL) + coreset
5. Gradient Manipulation: the presented paper
   - Orthogonal Weights Modification (OWM)

Context-Dependent Gating (XdG) [2]

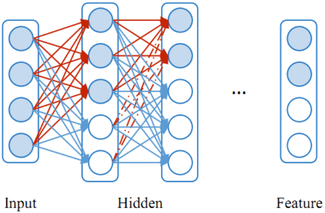
Key idea: to assign for each task a group of binary masks to gate/zero-out $X\%$ (*e.g.*, 80%) hidden units.



- The specified-then-fixed masks will be used in testing
- XdG only applies to Task-IL.

[2] Nicolas Y Masse, Gregory D Grant, and David J Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. PNAS, 2018.

Progress & Compress (P&C) [3]



Key Idea: Features are fixed once trained on previous tasks; latter tasks only use those features but do not update them.

- Ideally, latter tasks won't affect model performance on previous ones.
- New nodes will be added if necessary.

[3] Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y.W., Pascanu, R. and Hadsell, R. Progress & Compress: A scalable framework for continual learning. ICML, 2018.

A Rough Summary of Existing Methods

1. Naive: Exact Replay & Coreset
2. Model Architecture Manipulation
   - Context-Dependent Gating (XdG): Divide model capacity
   - Progress & Compress (P&C): Increase model capacity
3. **Replay-Based: Replay previous ability and/or previous data**
   - Learning without Forgetting (LwF)
   - Deep Generative Replay (DGR) & Distillation
   - Incremental Classifier and Representation Learning (iCaRL) + coreset
4. Regularization: Evaluate parameter importance
   - Elastic Weight Consolidation (EWC)
   - Online EWC
   - Synaptic Intelligence (SI)
   - *Variational Continual Learning (VCL) + coreset
5. Gradient Manipulation: the presented paper
   - Orthogonal Weights Modification (OWM)

Learning without Forgetting (LwF) [4]

Key: to distill previous model ability with **current data.**

$$\mathcal{L}_{total} = \frac{1}{N_{TaskSoFar}} \mathcal{L}_{current} + (1 - \frac{1}{N_{TaskSoFar}}) \mathcal{L}_{distill}. \qquad (1)$$

Given the current data $\{\boldsymbol{x}, y\}$ from Task K and model $p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})$,

$$\mathcal{L}_{current} = \text{CrossEntropy}(p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x}), y) \qquad (2)$$

and, with a copy of previous $p^T_{\hat{\boldsymbol{\theta}}^{(K-1)}}(y|\boldsymbol{x})$ with temperature T,

$$\mathcal{L}_{distill} = \text{SoftCrossEntropy}(p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x}), p^T_{\hat{\boldsymbol{\theta}}^{(K-1)}}(\cdot|\boldsymbol{x})) \times T^2. \qquad (3)$$

Note before softmax, the logits are divided by T.

[4] Zhizhong Li and Derek Hoiem. Learning without forgetting. TPAMI, 2017.

Deep Generative Replay (DGR) [5] + Distillation

Key: to replay previous observed data
DGR:

1. Train a generative model to replay previous images $\{x'\}$;
2. use a copy of the previous classifier to generate (hard) pseudo labels $\{y'\}$;
3. combine the replayed data $\{x', y'\}$ with the current data $\{x, y\}$ to train the current classifier.

DGR+Distill: Use the soft pseudo labels (probabilities) and knowledge distillation following LwF.

[5] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. NeurIPS, 2017.

Incremental Classifier and Representation Learning (iCaRL) [6] Replay/Distillation + Coreset

Key Idea: Feature domain classification aided by B exemplars.

- A universal feature extractor $\psi_{\boldsymbol{\phi}}(\boldsymbol{x})$; $m = \lfloor \frac{B}{N_{ClassSoFar}} \rfloor$ examples $\{\boldsymbol{p}_i\}_{i=1}^{m}$ per class. Class-IL

Training. With $\boldsymbol{\theta} = \{\boldsymbol{\phi}, \{\boldsymbol{w}_c\}\}$, $\boldsymbol{w}_c$ the parameters of class c,

$$\mathcal{L}_{iCaRL}(\boldsymbol{\theta}) = -\sum \left[\bar{y} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) + (1-\bar{y}) \log(1 - p_{\boldsymbol{\theta}}(\boldsymbol{x}))\right], \quad (4)$$

where “old-task-soft-target/new-task-hard-target” (Distillation)

$$\bar{y} = \begin{cases} p^{c}_{\hat{\boldsymbol{\theta}}^{(K-1)}}(\boldsymbol{x}) & \text{if } c \in \{1, \cdots, K-1\} \\ \delta_{y=c} & \text{if } c = K \end{cases} \quad (5)$$

[6] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. iCaRL: Incremental classifier and representation learning. CVPR, 2017.

Incremental Classifier and Representation Learning (iCaRL) (Continued)

After training, top m observations, keeping the closest feature mean, are greedily selected as exemplars for Task K.

Testing. Nearest-Class-Mean Classification. Given $\boldsymbol{x}$, its label

$$y^* = \underset{c}{\operatorname{argmin}} \|\psi_{\boldsymbol{\phi}}(\boldsymbol{x}) - \boldsymbol{\mu}_c\|, \tag{6}$$

where $\boldsymbol{\mu}_c = \frac{1}{|\mathcal{P}_c|} \sum_{\boldsymbol{p} \in \mathcal{P}_c} \psi_{\boldsymbol{\phi}}(\boldsymbol{p})$ is the feature mean of class c.

A Rough Summary of Existing Methods

1. Naive: Exact Replay & Coreset
2. Model Architecture Manipulation
   - Context-Dependent Gating (XdG): Divide model capacity
   - Progress & Compress (P&C): Increase model capacity
3. Replay-Based: Distill previous ability/replay previous data
   - Learning without Forgetting (LwF)
   - Deep Generative Replay (DGR) & Distillation
   - Incremental Classifier and Representation Learning (iCaRL) + coreset
4. **Regularization: Evaluate parameter importance**
   - Elastic Weight Consolidation (EWC)
   - Online EWC
   - Synaptic Intelligence (SI)
   - *Variational Continual Learning (VCL) + coreset
5. Gradient Manipulation: the presented paper
   - Orthogonal Weights Modification (OWM)

Elastic Weight Consolidation (EWC) [7]

$$\mathcal{L}_{total} = \mathcal{L}_{current} + \lambda \mathcal{L}_{reg} \tag{7}$$

For Task $K > 1$,

$$L_{reg_{\text{EWC}}}^{(K)}(\boldsymbol{\theta}) = \frac{1}{2} \sum_{k=1}^{K-1} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}^{(k)})^T \hat{\mathbf{F}}^{(k)} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}^{(k)}), \tag{8}$$

where $\hat{\mathbf{F}}^{(k)}$ is the diagonal of the Fisher information $\mathbf{F}^{(k)}$ of Task k with definition

$$\mathbf{F}^{(k)} = \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})} \left[[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})][\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})]^T \right] \Big|_{\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}^{(k)}}. \tag{9}$$

Sample-based approximation is often used.

[7] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, et al. Overcoming catastrophic forgetting in neural networks. PNAS, 2017.

Online EWC

Motivations: Maintaining $\{\hat{\boldsymbol{\theta}}^{(k)}\}_{k=1}^{K-1}$ (EWC) is expensive. For Task $K > 1$,

$$L_{reg_{\text{oEWC}}}^{(K)}(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}^{(K-1)})^T \tilde{\mathbf{F}}^{(K-1)}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}^{(K-1)}), \quad (10)$$

where $\tilde{\mathbf{F}}^{(K-1)}$ is a running sum of the diagonal of previous Fisher information, *i.e.*,

$$\tilde{\mathbf{F}}^{(K)} = \gamma \tilde{\mathbf{F}}^{(K-1)} + \hat{\mathbf{F}}^{(K)}, \quad (11)$$

where $\hat{\mathbf{F}}^{(K)}$ is the diagonal approximation of the Fisher information $\mathbf{F}^{(K)}$ of Task K.

Synaptic Intelligence (SI) [8]

$$L^{(K)}_{reg_{\mathrm{SI}}}(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}^{(K-1)})^T \boldsymbol{\Omega}^{(K-1)} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}^{(K-1)}), \tag{12}$$

where $\boldsymbol{\Omega}^{(K-1)}$ is a diagonal matrix with element

$$\boldsymbol{\Omega}^{(K-1)}_{ii} = \sum_{k=1}^{K-1} \frac{\omega_i^{(k)}}{[\Delta_i^{(k)}]^2 + \xi}, \tag{13}$$

$$\omega_i^{(k)} = \sum_{t=1}^{N_{iter}} (\theta_i[t] - \theta_i[t-1]) \left[-\nabla_{\theta_i} \mathcal{L}^{(k)}_{total}[t] \right], \tag{14}$$

where N_{iter} is the total number of iterations per task and $\Delta_i^{(k)} = \theta_i[N_{iter}] - \theta_i[0]$.

[8] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. ICML, 2017.

A Rough Summary of Existing Methods

1. Naive: Exact Replay & Coreset
2. Model Architecture Manipulation
   - Context-Dependent Gating (XdG): Divide model capacity
   - Progress & Compress (P&C): Increase model capacity
3. Replay-Based: Distill previous ability/replay previous data
   - Learning without Forgetting (LwF)
   - Deep Generative Replay (DGR) & Distillation
   - Incremental Classifier and Representation Learning (iCaRL) + coreset
4. Regularization: Evaluate parameter importance
   - Elastic Weight Consolidation (EWC)
   - Online EWC
   - Synaptic Intelligence (SI)
   - *Variational Continual Learning (VCL) + coreset
5. **Gradient Manipulation: the presented paper**
   - Orthogonal Weights Modification (OWM)

Comparisons on SplitMNIST & PermuteMNIST

Approach	Method	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower bound*	*87.19 (± 0.94)*	*59.21 (± 2.04)*	*19.90 (± 0.02)*
	Offline – upper bound	*99.66 (± 0.02)*	*98.42 (± 0.06)*	*97.94 (± 0.03)*
Task-specific	XdG	99.10 (± 0.08)	-	-
Regularization	EWC	98.64 (± 0.22)	63.95 (± 1.90)	20.01 (± 0.06)
	Online EWC	99.12 (± 0.11)	64.32 (± 1.90)	19.96 (± 0.07)
	SI	99.09 (± 0.15)	65.36 (± 1.57)	19.99 (± 0.06)
Replay	LwF	99.57 (± 0.02)	71.50 (± 1.63)	23.85 (± 0.44)
	DGR	99.50 (± 0.03)	95.72 (± 0.25)	90.79 (± 0.41)
	DGR+distill	99.61 (± 0.02)	96.83 (± 0.20)	91.79 (± 0.32)
Replay + Exemplars	iCaRL (budget = 2000)	-	-	94.57 (± 0.11)

↑ SplitMNIST ↓ PermuteMNIST

Approach	Method	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower bound*	*81.79 (± 0.48)*	*78.51 (± 0.24)*	*17.26 (± 0.19)*
	Offline – upper bound	97.68 (± 0.01)	97.59 (± 0.01)	97.59 (± 0.02)
Task-specific	XdG	91.40 (± 0.23)	-	-
Regularization	EWC	94.74 (± 0.05)	94.31 (± 0.11)	25.04 (± 0.50)
	Online EWC	95.96 (± 0.06)	94.42 (± 0.13)	33.88 (± 0.49)
	SI	94.75 (± 0.14)	95.33 (± 0.11)	29.31 (± 0.62)
Replay	LwF	69.84 (± 0.46)	72.64 (± 0.52)	22.64 (± 0.23)
	DGR	92.52 (± 0.08)	95.09 (± 0.04)	92.19 (± 0.09)
	DGR+distill	97.51 (± 0.01)	97.35 (± 0.02)	96.38 (± 0.03)
Replay + Exemplars	iCaRL (budget = 2000)	-	-	94.85 (± 0.03)

Table of Contents

Orthogonal Weights Modification (OWM)

Key Idea: to project gradient to preserve model capabilities.

- For simplicity, consider linear regression first
- In previous task, we have data matrices $\{\mathbf{A}_0, \mathbf{B}_0\}$ and the optimal $\mathbf{W}_0$ satisfying $\mathbf{B}_0 = \mathbf{W}_0^T \mathbf{A}_0$;
- For the current task with data $\{\mathbf{A}_1, \mathbf{B}_1\}$, how to train $\mathbf{W}_1$ to satisfy $\mathbf{B}_1 = \mathbf{W}_1^T \mathbf{A}_1$ with the constraint $\mathbf{B}_0 = \mathbf{W}_1^T \mathbf{A}_0$?

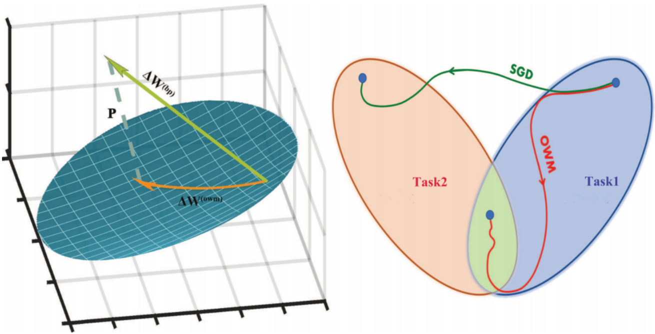
OWM projects the gradient $\Delta\mathbf{W}$ $(= \nabla_{\mathbf{W}} ||\mathbf{B}_1 - \mathbf{W}^T \mathbf{A}_1||)$ to make sure $\mathbf{B}_0 = \mathbf{W}^T \mathbf{A}_0$ is always satisfied via

$$\mathbf{W}' = \mathbf{W}_0 - \eta \underbrace{\left[\mathbf{I} - \mathbf{A}_0(\mathbf{A}_0^T \mathbf{A}_0)^{-1} \mathbf{A}_0^T\right]}_{\mathbf{P}} \Delta\mathbf{W} \tag{15}$$

$\mathbf{P}$ can be approximated first and then calculated **recursively**

$$\begin{aligned} \mathbf{P} &\approx \mathbf{I} - \mathbf{A}(\alpha\mathbf{I} + \mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \\ &= \alpha \left[\alpha\mathbf{I} + \mathbf{A}\mathbf{A}^T\right]^{-1} = \alpha \left[\alpha\mathbf{I} + \sum_i \boldsymbol{x}_i \boldsymbol{x}_i^T\right]^{-1} \end{aligned} \tag{16}$$

Orthogonal Weights Modification (OWM) (continued)

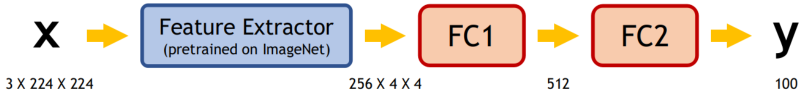


- For lifelong/continual learning of deep neural networks, apply OWM to each linear layer.

Table of Contents

OWM Aided by Pretrained Feature Extractors

Assume a fully developed feature extractor is available



Data Set	Classes	Feature Extractor	Concurrent Training by SGD (%)	Sequential Training by OWM (%)	Sequential Training by SGD (%)
ImageNet	1000	ResNet152	78.31	75.24	4.27
CASIA-HWDB1.1	3755	ResNet18	97.46	93.46	35.86

Analogous to humans' learning in cognition

- Humans can easily form new concepts of objects Class-IL
- with fully developed sensory cortices (feature extractors).
  - may take years or even decades

*On Pretraining the Feature Extractor (CASIA-HWDB)

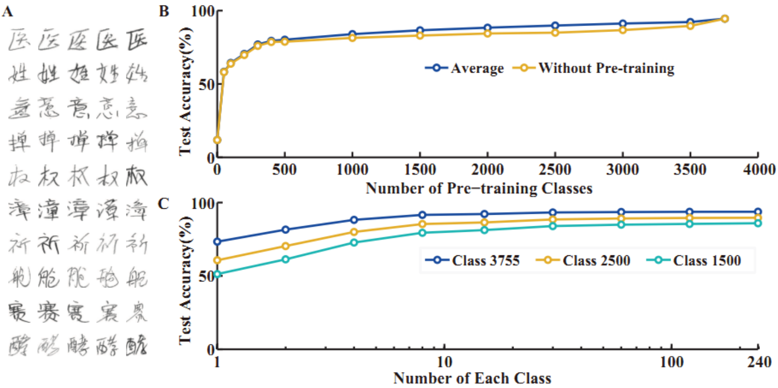
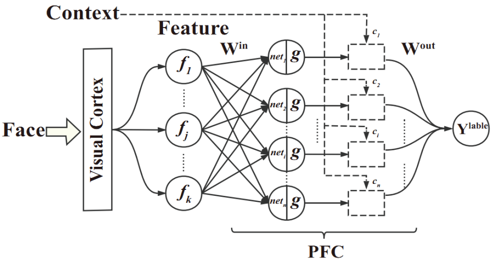


Figure 2. Online learning with small sample size achieved by OWM in recognizing Chinese characters. **(A)** Examples showing 10 characters with five samples for each. **(B)** Classification accuracy is plotted as a function of the number of classes used for pre-training the feature extractor. The performance was assessed based on classifying all characters (blue) or characters that were not included in the pre-training (orange). **(C)** Classification accuracy is plotted as a function of the sample size used for sequential training, obtained with feature extractors having different degrees of pre-training (color-coded).

On Incorporating Contextual Information (CelebA)

Situations: Same input, but different processing based on different context

- CelebA: faces with 40 attributes (contextual information)
- Male, Wear lipstick, Mouth small open, Attractive, ...



$$\boldsymbol{y}_{PFC}^{out} = g((\mathbf{W}^{in})^T \boldsymbol{f}) \odot \boldsymbol{c} \tag{17}$$

$\mathbf{W}^{in}$: randomly initialized & column-normalized. $\boldsymbol{c}$: uniformly generated for each attribute/context. $\mathbf{W}^{out}$: trained with OWM.

On Incorporating Contextual Information (CelebA) (Continued)

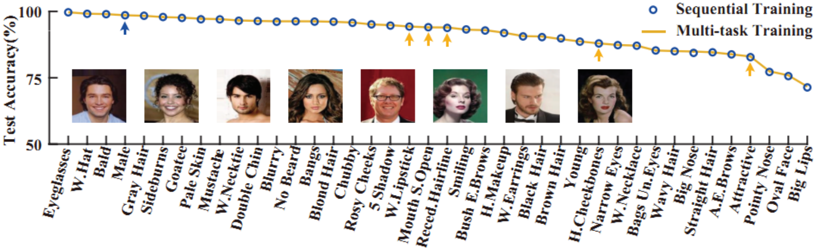


Table of Contents

Concluding Remarks

- General lifelong learning is extremely challenging,
  - especially when considering vast practical situations.
- For special cases with clear task boundaries, to remember previous data/sufficient-information might be a good idea.
  - **Data:** Coreset/Exemplars or generative-replay
  - **Sufficient-Information:** To find a tricky cheap way to collect such information (like an OWM projection matrix)
- To train a universal feature extractor, a good idea might be dynamic model architecture with increasing nodes but with fixed previous features.
- General lifelong learning needs combinations of existing ideas.