PODNet: Pooled Outputs Distillation for Small-Tasks Incremental Learning

Mario De Simone

Faculty of Artificial Intelligence
University of Florence

June 2025

Table of Contents

Introduction

Class-Incremental Learning (CIL) is a branch of a wider **Incremental Learning** framework, which purpose is to increase knowledge and ability of a model in different 'steps'

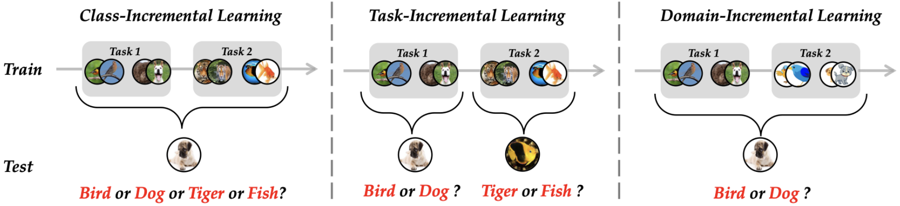


Figure: Different settings of Incremental Learning

CIL and Catastrophic Forgetting

The worst enemy for CIL is **Catastrophic Forgetting**, the model overwrites past knowledge learning new classes

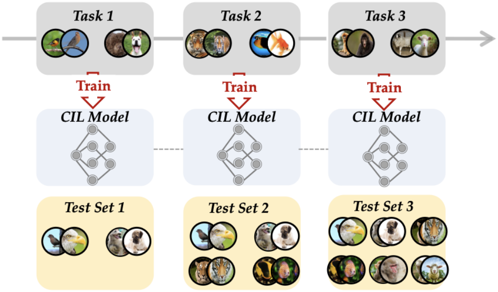


Figure: CIL framework

Other Works

Taxonomy suggested by [4]:

- **Data Replay**: there is a memory to store past examples
- **Data Regularization**: control of the optimization process in order to avoid hurting past classes
- **Dynamic Networks**: dynamically adjust the weights, in order to learn new classes and not forget previous ones
- **Parameter Regularization**: find and fix the most important part of the network and keep it static
- **Knowledge Distillation**: learning in steps, each one uses the knowledge of the previous one to resist forgetting
- **Model Rectify**: correct the behavior of CIL, trying to reach ideal 'oracle'
- **Template-based Classification**: try to build a template (prototype) representing a class

PODNet Roots

PODNet is a model suggested by [1]:

- **Replay Memory**: the network takes into account the usage of a part of memory for storing past examples
- **Distillation Knowledge**: the paper suggests a new distillation loss with the aim of reducing catastrophic forgetting
- **Template-based Classification**: a new kind of classifier is employed in order to reach a shift-invariant representation

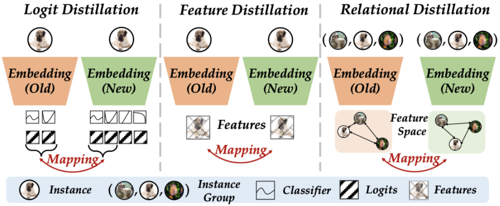


Figure: Example of a Knowledge Distillation framework

PODNet Intro

Some notation about incremental learning:

- T tasks
- C_t^N set of new classes (to learn) of task t
- $C_t^O = C_{t-1}^N \cup C_{t-1}^O$ set of old classes (already seen) of task t
- C_t^O is a **limited memory** M_{per} samples per class

Moreover classic deep neural network frame with:

$$\hat{y} = g(h) \quad where \quad h = f(x)$$

h are the features extracted and g the classification layer

Rigidity vs Plasticity

Two key concepts control incremental learning:

- **Rigidity**: ability to resist to variations
- **Plasticity**: oppose of rigidity

It is easy to suppose there is a trade-off between the two, we have to optimize in order to reach good performances

POD

Pooled Outputs Distillation (POD) is a set of constraints using the invariance of pooling to intermediate layers (h_l^t features of layer l of task t):

$$L_{POD-width}(h_l^{t-1}, h_l^t) = \sum_{c=1}^{C} \sum_{h=1}^{H} \| \sum_{w=1}^{W} h_{l,c,w,h}^{t-1} - \sum_{w=1}^{W} h_{l,c,w,h}^{t} \|^2$$

$$L_{POD-height}(h_l^{t-1}, h_l^t) = \sum_{c=1}^{C} \sum_{w=1}^{W} \| \sum_{h=1}^{H} h_{l,c,w,h}^{t-1} - \sum_{h=1}^{H} h_{l,c,w,h}^{t} \|^2$$

$$L_{POD-spatial}(h_l^{t-1}, h_l^t) = L_{POD-width}(h_l^{t-1}, h_l^t) + L_{POD-height}(h_l^{t-1}, h_l^t)$$

Due to the flattening of the last layer the POD idea must be adjusted:

$$L_{POD-flat}(h_l^{t-1}, h_l^t) = \| h^{t-1} - h^t \|^2$$

Multimodal Classification Layer

To fight shift in the distribution of h, the classifiers learns K vectors (modes) for each class:

$$\hat{y}_c = \sum_k s_{c,k} \langle \theta_{c,k}, h \rangle \quad where \quad s_{c,k} = \frac{\exp(\langle \theta_{c,k}, h \rangle)}{\sum \exp(\langle \theta_{i,k}, h \rangle)}$$

moreover to fight the imbalance of data [2] proposed cosine normalization, so in the end the classification loss become:

$$L_{lsc} = \left[-\log \frac{\exp(\eta(\hat{y}_y - \delta))}{\sum_{y \neq i} \exp \eta \hat{y}_i} \right]_+$$

Distillation Loss

Putting all together we get:

$$L_{POD-final} = \frac{\lambda_c}{L-1} \sum_{l=1}^{L-1} L_{POD-spatial}(f_l^{t-1}(x), f_l^t(x)) + \lambda_f L_{POD-flat}(f^{t-1}(x), f^t(x))$$

Taking into account the classification loss too, we arrive to the total loss:

$$L_{total} = L_{POD-final} + L_{lcs}$$

- L is the total number of layers
- λ_c and λ_f act as importance weights for the two POD losses
- the feature of $POD_spatial$ are l2 normalized
- [2] suggests to scale pod loss by a factor $\lambda = \sqrt{\frac{N}{T}}$ where N is the number of classes already seen and T the number of classes in the current task

PODNet Architecture

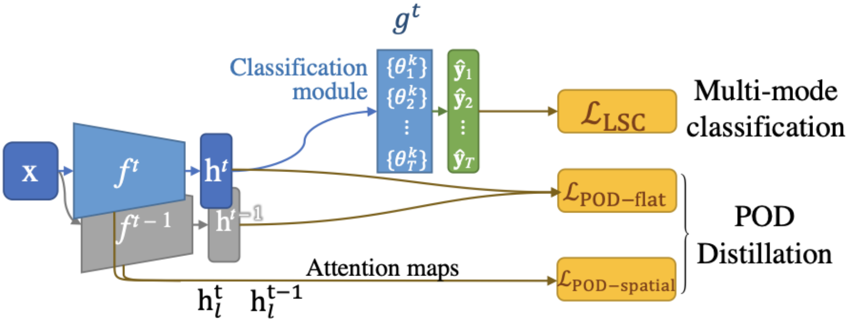


Figure: PODNet Architecture

Experiments Details

- **Optimizer**: SGD with learning rate 0.1 and 0.8 momentum
- **Batch Size**: 128
- **Metric**: Average Incremental Accuracy (suggested by [3])

	Cifar100	Imagenet(100-1000)
Epochs	160	90
Weight Decay	$5 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
(λ_c, λ_f)	$(3, 1)$	$(8, 10)$

Table: Experiments Settings

Imagenet Results

	ImageNet100				Imagenet1000	
	50 steps	25 steps	10 steps	5 steps	10 steps	5 steps
New classes per step	1	2	5	10	50	100
iCaRL* [33]	—	—	59.53	65.04	46.72	51.36
iCaRL [33]	54.97	54.56	60.90	65.56	—	—
BiC [38]	46.49	59.65	65.14	68.97	44.31	45.72
UCIR (NME)* [14]	—	—	66.16	68.43	59.92	61.56
UCIR (NME) [14]	55.44	60.81	65.83	69.07	—	—
UCIR (CNN)* [14]	—	—	68.09	70.47	61.28	64.34
UCIR (CNN) [14]	57.25	62.94	67.82	71.04	—	—
PODNet (CNN)	**62.48**	**68.31**	**74.33**	**75.54**	**64.13**	**66.95**
	± 0.59	**± 2.45**	**± 0.93**	**± 0.26**		

Figure: Average Incremental Accuracy on Imagenet datasets

Cifar100

	CIFAR100			
	50 steps	25 steps	10 steps	5 steps
New classes per step	1	2	5	10
*iCaRL** [33]	—	—	52.57	57.17
iCaRL	44.20 ± 0.98	50.60 ± 1.06	53.78 ± 1.16	58.08 ± 0.59
BiC [38]	47.09 ± 1.48	48.96 ± 1.03	53.21 ± 1.01	56.86 ± 0.46
*UCIR (NME)** [14]	—	—	60.12	63.12
UCIR (NME) [14]	48.57 ± 0.37	56.82 ± 0.19	60.83 ± 0.70	63.63 ± 0.87
*UCIR (CNN)** [14]	—	—	60.18	63.42
UCIR (CNN) [14]	49.30 ± 0.32	57.57 ± 0.23	61.22 ± 0.69	64.01 ± 0.91
PODNet (NME)	$\mathbf{61.40 \pm 0.68}$	$\mathbf{62.71 \pm 1.26}$	$\mathbf{64.03 \pm 1.30}$	$\mathbf{64.48 \pm 1.32}$
PODNet (CNN)	$\mathbf{57.98 \pm 0.46}$	$\mathbf{60.72 \pm 1.36}$	$\mathbf{63.19 \pm 1.16}$	$\mathbf{64.83 \pm 0.98}$

Figure: Average Incremental Accuracy on Cifar100 dataset

Remotion $POD_{spatial}$

Classifier	POD-flat	POD-spatial	NME	CNN
Cosine			40.76	37.93
Cosine	✓		48.03	46.73
Cosine		✓	54.32	57.27
Cosine	✓	✓	56.69	55.72
LSC-CE	✓	✓	59.86	57.45
LSC			41.56	40.76
LSC	✓		53.29	52.98
LSC		✓	**61.42**	57.64
LSC	✓	✓	61.40	**57.98**

Figure: Average Incremental Accuracy disabling parts of the model

Alternatives to $POD_{spatial}$

Loss	NME	CNN
None	53.29	52.98
POD-pixels	49.74	52.34
POD-channels	57.21	54.64
POD-gap	58.80	55.95
POD-width	60.92	57.51
POD-height	60.64	57.50
POD-spatial	**61.40**	**57.98**
GradCam [5]	54.13	52.48
Perceptual Style [16]	51.01	52.25

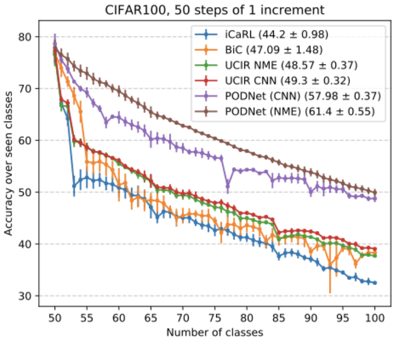
Figure: Average Incremental Accuracy changing the distillation method of intermediate layers

Remotion POD_{flat}

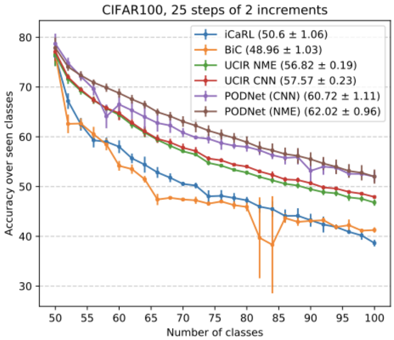
Loss	NME	CNN
None	41.56	40.76
POD-pixels	42.21	40.81
POD-channels	55.91	50.34
POD-gap	57.25	53.87
POD-width	61.25	57.51
POD-height	61.24	57.50
POD-spatial	**61.42**	**57.64**
GradCam [5]	41.89	42.07
Perceptual Style [16]	41.74	40.80

Figure: Effect of the remotion of POD_{flat}

Different Steps



(a) 50 steps, 1 class / step



(b) 25 steps, 2 classes / step

Figure: Performance of various model changing 'step-size'

Different Memory Size

M_{per}	5	10	**20**	50	100	200
iCaRL [33]	16.44	28.57	44.20	48.29	54.10	57.82
BiC [38]	20.84	21.97	47.09	55.01	62.23	**67.47**
UCIR (NME) [14]	21.81	41.92	48.57	56.09	60.31	64.24
UCIR (CNN) [14]	22.17	42.70	49.30	57.02	61.37	65.99
PODNet (NME)	**48.37**	**57.20**	**61.40**	**62.27**	**63.14**	63.63
PODNet (CNN)	**35.59**	**48.54**	**57.98**	**63.69**	**66.48**	**67.62**

Figure: Evaluation on different models changing M_{per}

Conclusion

To conclude the model suggested by [1] outperform state-of-art architectures thanks to:

- a multimodal classifier avoiding catastrophic forgetting
- a smart pooling method aggregating spatial features at different layers
- a distillation loss solving the rigidity vs plasticity trade-off

Some research ideas could be:

- make a transition to a no-replay memory model
- find a way to select the most 'significant' M_{per} *examples*
- find a way to weight features at different layers

References

[1] Arthur Douillard et al. *PODNet: Pooled Outputs Distillation for Small-Tasks Incremental Learning*. 2020.

[2] Saihui Hou et al. "Learning a Unified Classifier Incrementally via Rebalancing". In: 2019.

[3] Sylvestre-Alvise Rebuffi et al. *iCaRL: Incremental Classifier and Representation Learning*. 2017.

[4] Da-Wei Zhou et al. "Class-Incremental Learning: A Survey". In: (2024).