Research Project Carried Out at Etis Lab

Data Incremental Learning in Deep Architectures

by

ELIE ALEX KAMENI

Master Student in DSDM

ETIS UMR8051, University of CY Cergy Paris / ENSEA / CNRS
6 avenue du Ponceau, 95014 Cergy-Pontoise Cedex, France

Defended September, 2022

In front of the jury composed of :

Pr.	Son VU	CY Cergy Paris Universite	In charge of the intership
Pr.	xxx	ENSEA	Reporter and Professor advisor

Data Incremental Learning in Deep Architectures

Alex KAMENI
ETIS
Cergy, France
elie-alex.kameni-ngangue@ensea.fr

Vu SON
ETIS
Cergy, France
son.vu@ensea.fr

***Abstract*—Without relying on human annotations, self-supervised learning aims to learn useful representations of input data. When trained offline with enormous amounts of unlabelled data, self-supervised models have been found to provide visual representations that are equivalent to or better than supervised models. However, in continual learning (CL) circumstances, when data is fed to the model sequentially, its efficacy is drastically diminished. Numerous ongoing learning techniques have recently been presented for a variety of computer vision problems. In this study, by utilizing distillation and proofreading, we tackle the extremely challenging problem of continuously learning a usable representation in which input data arrives sequentially. We can prevent severe forgetfulness and continue to train our models by adding a prediction layer that forces the current representations vectors to precisely match the frozen learned representations and an effective selection memory for proofreading previous data. This makes it possible for us to design a framework for continual self-supervised learning of visual representations that (i) greatly enhances the quality of the learned representations, (ii) is suitable for a number of state-of-art self-supervised objectives, and (iii) requires little to no hyperparameter tuning. The code of this paper is made available here cssl-dsdm**

I. Introduction

Deep learning architectures have achieved outstanding improvements in computer vision, but when forced to learn new tasks progressively without forgetting the old, their performance suffers significantly. The use of artificial intelligence in situations where computers must gradually acquire new and varied representations is impacted by this catastrophic forgetting issue.

The most well-liked paradigm for unsupervised visual representation learning is self-supervised learning (SSL). Under some conditions (such as offline training with lots of data and resources), SSL techniques can extract representations with a similar level of quality as those discovered by supervised learning. In real-world situations, such as when additional unlabeled data is gradually made accessible over time, these assumptions may not always be true. In fact, training must be repeated across the entire dataset in order to incorporate new knowledge into the model, which is unnecessary, expensive, and occasionally even impossible when the old data is unavailable. Continuous learning is one solution to this issue, where models are updated gradually as new data comes in.

Continual learning is a paradigm for learning where various data and tasks are supplied to the model sequentially, much to what humans often experience. Artificial neural networks (ANNs) prefer learning in a more concurrent way and have been demonstrated to forget catastrophically, in contrast to human or animal learning, which is mostly gradual and sequential in nature. The failure of ANNs to remember past knowledge in the presence of fresh input is typically referred to as catastrophic forgetting (CF) in neural networks. Catastrophic forgetting is a direct implication of continual learning in ANNs and is largely considered as a direct consequence of the overlap of distributed representations in the network. The commonly used definition of continual learning refers to a paradigm for learning where ANNs are rigidly trained sequentially on various datasets and tasks ([1, 2]). The important conditions of the training paradigm are:

- Sequential training i.e. for a single neural network f with parameters θ trained at time T with sequentially available data $D_{1...N}$,
$$T_1(f_\theta(D_1)) < T_1(f_\theta(D_1)) < ... < T_1(f_\theta(D_1)) \quad (1)$$
- No negative exemplars, examples or feedback i.e. future (or past) data samples cannot be provided to the network with the current data/task,
$$\forall_{i,j} \in (1...N), (D_i \cap D_j)_{i \neq j} = \emptyset \quad (2)$$

Most prior works deal with CF by either completely removing the representational overlap ([3, 4]) or more frequently, by replaying data from previous tasks. Data replay methods can deal with CF but, in turn, lead to a reduced capability of the network to discriminate between old and new inputs ([5]). This is referred to as Catastrophic Remembering (CR) and has been shown to be a significant limitation of replay methods ([6, 5]).

Regardless of the method used, continual learning problems can be categorized into task-incremental learning (Task-IL), domain-incremental learning (Domain-IL) and class-increasing learning (Class-IL). In the first scenario, models are always informed about which task needs to be performed. In the second scenario, task identity is not available at test time. Models however only need to solve the task at hand; they are not required to infer which task it is. Finally, in the third scenario models must be able to both solve each task seen so far and infer what task they are presented with. Class incremental learning is the most complex case of data incremental learning where classes in two data are different.

Continual learning issues can be divided into task-incremental learning (Task-IL), domain-incremental learning (Domain-IL), and class-increasing learning (Class-IL), depending on the approach taken. Models in the first scenario are always aware of the work that has to be achieved. The task identification is unavailable at test time in the second case. However, models are only necessary to complete the current work; they are not required to determine what the current task is. In the third scenario, models must be able to both complete each task that has been shown so far and determine the task that is being offered to them. The most challenging instance of data incremental learning occurs when classes in two data sets differ.

The goal of this work is that we attempt to develop a method for data incremental continual learning for deep neural networks which can alleviate the twin problem of Catastrophic Forgetting and Catastrophic Remembering at the same time, without violating or relaxing the conditions of a strict continual learning framework.

Common benchmarks in the CL literature evaluate the discriminative performance of classifiers learned with supervision from non-stationary distributions. In this work, we follow the same forgetting phenomenon in the context of SSL.

II. Related work

A. Self-Supervised Learning

Modern deep neural networks may be taught to perform impressively on a range of different individual tasks. Deep learning still faces significant difficulties with sequentially learning several tasks, nevertheless.Standard neural networks experience a problem known as ”catastrophic forgetting” when they mostly forget the knowledge associated with previously acquired tasks when trained on a new task.

Without the use of human annotations, self-supervised learning attempts to develop meaningful representations of the incoming data. Recent developments in this area demonstrate that representations that are learned can be competitive with supervised representations. These techniques employ various distorted samples, asymmetric learning updates, momentum encoders, and non-differentiable operators, among others. Approaches for self-supervised learning (SSL) have demonstrated performance on par with supervised learning counterparts [7, 8, 9, 10, 11, 12]. The majority of these approaches produce associated views (positives) from a sample using picture augmentation techniques, after which they build a model that is resistant to the augmentations. To avoid simple solutions, the learning update’s asymmetry, or ”stop-gradients,” is essential.

At the same time, cluster-based methods (SwAV [13], DeepCluster [14], and DINO [9]) were also proposed. They do not manipulate features directly, but instead use cluster prototypes as proxies to compare positive results by cross-entropy loss. Methods based on redundancy reduction were also popular. Among these, BarlowTwins [12] considers an objective function that measures the cross-correlation matrix between features, and VicReg [7] uses a combination of variance, invariance, and covariance regularization. Methods such as [15] explore the use of nearest neighbour search and divide-and-conquer [16]. However, none of the studies examined the ability of SSL methods to learn continually and adaptively.

B. Continual Learning Setting

Continual learning has been studied in a variety of scenarios, often grouped into three categories of increasing difficulty: tasks, domains, and class increments [17]. Instead of using a single-head classifier for all classes as class increment setting, the task increment problem method uses a multi-head classifier for each independent task and the domain increment method is intended to learn label shifts rather than new classes.

Tasks, domains, and class increments are three categories of increasing difficulty that have all been used to study continual learning [17]. The task increment problem technique employs a multi-head classifier for each independent job instead of utilizing a single-head classifier for all classes like the class increment setting, and the domain increment method is designed to learn label changes rather than new classes.

Continual learning may be divided into online learning, where all data is utilized just once, and offline learning without epoch restrictions, depending on whether all data is used more than once to update the model. Furthermore, depending on whether the boundaries between various tasks are known, continual learning can be task-based or task-free. A concept of incremental data learning has just recently developed [18] enables learning from each data stream without relying on presumptions on task IDs, task boundaries, or the chronological order of data observation.

This work studies the challenging scenario of continual learning under data incremental, offline setting where data arrives sequentially and our model tries to learn new features without forgetting previous learned representation.

C. Continual learning algorithms

Three main streams of techniques to continual learning have been developed: regularization-based, architecture-based, and memory-based (or rehearsal-based).

1) Regularization: This strategy seeks to retain each node’s plasticity while preserving the information gained from earlier nodes. Using the Fisher information matrix, Elastic Weight Consolidation (EWC) [4] estimates the weights’ relative contributions to earlier tasks. The significance of certain weights is thus taken into account when changing weights. Using a different regularization technique, each weight change is penalized based on a measurement of synaptic intelligence (SI) [19]. The task specificity of each parameter is estimated, and modifications to parameters with high task specificity are penalized. This guarantees that the parameter stays within a certain range of its current value, which was associated with successful learning of the initial task.

Knowledge Distillation (KD) is an efficient manner to transfer the knowledge between networks. It is initially introduced for network compression by using the teacher-student mechanism [20] and widely adopted for CL methods [21, 22] and is considered as one of regularization techniques. However, it is shown in [17] that methods based on strict regularization such as EWC and SI completely fail on class-IL setting. Instead of directly comparing the output of the current and frozen model, CaSSLe [23] proposes to project the running (current) into the same embedding space as frozen before comparing both representations. In this paper, we follow the same idea, but instead of projecting only the current task, we use a buffer to replay data from previous tasks, and we project both current and buffer data into their corresponding embedding space produced by the frozen model.

2) Architecture: These solutions are generally based on architectural alterations. As in [24], they add task-specific parameters that dynamically boost the model's capacity by including a new layer in the network. Other approaches of this nature dynamically build new network levels [25]. Other methods developed by [26, 27] try to freeze weights that the system determines to be crucial to a specific task and that won't be changed by subsequent back-propagation. Similar techniques employ a mask for every prior task to safeguard parameters during the backward pass or to select which parameters to utilize during the forward phase.

Since normalization layers are recognized to be crucial for training deep neural networks [28], recent research has focused particularly on this layer (CN [29]). A group norm followed by a batch norm is proposed as a replacement for BN, which normalizes testing data using moments skewed towards the present task and leads to greater rates of catastrophic forgetting. This would considerably lessen the detrimental effects of BN.

3) Memory-based approaches: The key component is to choose a strategy to store specific exemplars. It can be a strategy as simple as random selection or a more complex method, as in [30]. Some models can replay the exemplars stored in memory with the stream of data from the current task, or they can be used as a regularization term, as in GEM [31] or in A-GEM [32], by computing the scalar product of the loss gradient vector of previous examples stored in memory and the current data gradient vector. The parameters of the model are then only updated if the scalar product is positive. To overcome the necessity of an external memory to store data of previous tasks, as in memory-based, certain architectures integrate a generative model able to generate exemplars from previous tasks. These models include the use of a Generative Adversarial Network [33] as in [34] or a Variational AutoEncoder [35] as in [36]. These methods are also called Generative Replay methods. The main drawback is that generative models are themselves prone to catastrophic forgetting.

The most important step is to decide on a method for storing particular examples. It can take the form of a straightforward strategy like random selection or a sophisticated one like [30]. By computing the scalar product of the loss gradient vectors of previous examples stored in memory and the current data gradient vector, some models can replay the examples stored in memory with the stream of data from the current task, or they can be used as a regularization term, as in GEM [31] or in A-GEM [32]. Only when the scalar product is positive are the model's parameters updated. Some designs incorporate a generative model that may produce exemplars from prior task in order to avoid the need for an external memory to retain data from previous tasks, as in memory-based systems. These models employ either a Variational AutoEncoder [35] or a Generative Adversarial Network [33] similar to those used in [34] and [36]. These techniques are sometimes known as "generative replay" techniques. The primary flaw is that generative models are prone to catastrophic forgetting themselves.

This paper examines the recently proposed DSDM (Dynamic Sparse Distributed Memory) approach for class-incremental learning with an associative address-content memory that evolves dynamically and continuously to model the distribution of any non-stationary data streams. We also propose an effective memory selection that can be learned along with any unsupervised model and act as an efficient memory buffer.

III. Contributions

A. Preliminaries in SSL

Numerous state-of-the-art SSL approaches [7, 8, 9, 10, 12] can be summarized as follows. Two associated views, x_1 and x_2, obtained from an image x in a batch taken from a distribution D by using stochastic image augmentations such random cropping, colour jittering, and horizontal flipping. The view x_1 is fed to an encoder $f_\theta = f_p \circ f_b$ that extracts feature representations $z_1 = f_\theta(x_1)$ from its parametrized by backbone and projection head. To produce the representation z_2, x_2 is similarly forwarded into the same networks, or maybe replicas of them, and updated with an exponential moving average (EMA). To learn the parameters, a loss function L_{SSL} is used to these representations as follows:

$$argmin_\theta E_{x \sim D}[L_{SSL}(z_1, z_2)] \quad (3)$$

B. Distillation in CSSL

The goal of CSSL is to maximize the linear separability of features at the ends of the CL phase and to extract the best representations that can be used in a variety of tasks. Therefore, the stability of the representations does not provide much value to the linear classifier. This is because CSSL and supervised CL are fundamentally dissimilar. Additionally, restricting representations to remain static may impede the model from picking up new ideas. This is particularly important for SSL methods because (i) they exhibit different losses and feature normalizations that may interfere with distillation and vice versa, and (ii) their performance improves significantly with longer training, suggesting that the representations continue to get refined.

C. Proposed framework for CSSL

Our framework, shown in Figure 1 is composed of three main components : The rehearsal model, the current model and the frozen model.

We begin by creating a duplicate of the existing model whenever a new task arrives. This copy won't be changed because it doesn't need gradient calculation. This is referred to as the frozen encoder f^{t-1}. We use our stochastic image augmentations as soon as an image $x_1 \in D_t$ is available which

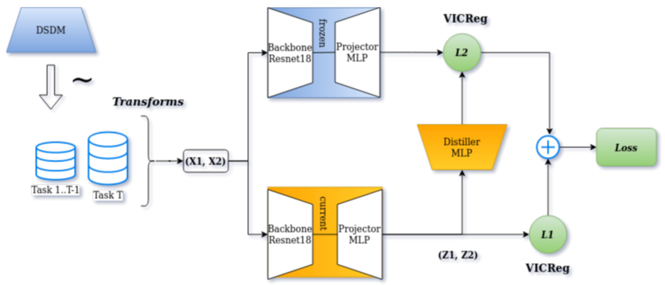

Fig. 1. Overview of proposed architecture

is appended to the data from our memory-based model, DSDM x_2 ($x = x_1 \cup x_2$) and we extract its features $z = f^t(x)$. Additionally, we extract another feature vector using the frozen encoder, $\tilde{z} = f^{t-1}(x)$. Now, we want to make sure that z has at least (and ideally) more information than $\tilde{z}$. We utilize a prediction network g to project the representations from the new feature space to the old one rather than constraining the two feature vectors to be comparable and preventing the new model from learning new ideas. It is implied that $\tilde{z}$ is at least as potent as z if the predictor can properly map from one space to the other [23].

We then define our loss as follows :

$$L(z, \tilde{z}) = L_{SSL}(z \sim (z_1, z2)) + L_D(z, \tilde{z}) \qquad (4)$$

Where L_{SSL} can be any of SSL loss like BarlowTwins [12], VICReg [7]... between distorted views and L_D is a compatible distillation loss between distorted views from current and frozen encoder.

D. Replay base model : DSDM

The recent online memory-based approach for task-free class incremental learning DSDM has shown promising result in continual learning. DSDM evolves dynamically and continually to model the distribution of any non-stationary data streams. It relies on locally distributed, but only partially overlapping clusters of representations to effectively eliminate catastrophic forgetting, while at the same time, maintaining the generalization capacities of distributed networks.

However, in order for this model to correctly model the distributions of the incoming data, it needs an encoder capable of performing an image-representation mapping. This is very problematic, because in a strict continuous learning context, this encoder will also have to be trained continually. We go through this limitation and propose to use a partially trained encoder to train the DSDM and the trained DSDM to perform replay.

When a new task arrives, we combine the data from this task with the data in our DSDM and start training our encoder. Once finished, we use this encoder and the data from the current task only to update our DSDM. At the end of this process, our DSDM model contains the features representative of the previous tasks and those of the current task.

Traditional DSDM begins with empty memory and gradually adds new neurons based on the input patterns being taken into account and the status of the memory space. In other words, DSDM develops new address nodes on an as-needed basis dependent on the input. This approach is inefficient and costly since it calls for iterating through each row of the dataset to compute representations, distances, and decide whether to build a new neuron in memory or update an existing one. In this study, we suggest processing the data in batches and switching from decision-making based on features to batch-based decision-making. This offers the benefit of creating a model that is considerably more reliable, fast, and effective.

IV. Experiments and results

A. Experiments settings

We have restricted our studies to BarlowTwins [12] and VICReg [7] techniques, the state-of-the-art in representation learning, due to the significant resources and time required to train an SSL encoder.

The datasets under consideration are CIFAR10 and CIFAR100, and a linear layer is used for evaluation of the three continual learning scenarios: task, data, and class-incremental learning.

Our memory model only keeps a tenth of the information from each prior task. 32 samples of DSDM memory are combined to a batch size of 128 from the current task for our SSL encoder during training, and finally a batch size of 256 is used for the training linear model. We trained the encoder 150 epochs just for each new task.

B. Evaluation protocol

We considered the three scenarios of continual learning :

- Task-IL : When we finish training on the current task, we evaluate using a linear classifier and the DSDM model the encoder on the task data. The score after this task is the average score of all classifiers trained before this task. It gives us an idea of the class separability of the current task.
- Class-IL : Unlike the previous scenario, this time we train our linear classifier to correctly distinguish all observed classes up to the current task. This score measures the linear separability of the encoder on all observed tasks.
- Data-IL : Unlike Class-IL and Task-IL, each subset contains all the classes of the base dataset. We thus measure the ability of our model to produce richer features when new data are made available.

C. Results and interpretations

1) Comparison of classic DSDM vs proposed F-DSDM: Table I shows a comparison of classic DSDM with the DSDM proposed in this paper in class-incremental learning scenario. In bold the best score on the current dataset. We have obtained significant improvement both CIFAR-10 and 100 considered.

2) Class-IL results: Table II shows results on CIFAR-10 and 100

TABLE I
CLASSIC DSDM VS PROPOSED F-DSDM

Datasets	CIFAR-10		CIFAR-100	
Accuracy	Last	Avg	Last	Avg
DSDM	76.0	85.6		63.3
F-DSDM	**85**	**90**		

TABLE II
CSSL FROM CASSLE VS REPLAY-BASED MODEL

	Dataset	CIFAR-10		CIFAR-100	
Methods	Linear Eval	Acc@1	Acc@5	Acc@1	Acc@5
BarlowTwins	Original	72.9	98.08	51	80
	Replay	**76.9**	**98.89**	**55**	**82.7**
VICReg	Original	71.8	97.96	51.5	78.5
	Replay	**72.88**	**98.08**	**53.23**	**81.4**

V. CONCLUSION

A. Conclusions on the presented work

B. Perspectives

ACKNOWLEDGMENT

This research was supported by the University of CY Cergy Paris/ENSEA. We thank our Professor "Vu SON" who provided insight and judgement on our oral presentation that greatly assisted in the making of this paper.

REFERENCES

[1] Zhiyuan Chen and B. Liu. "Lifelong Machine Learning, Second Edition". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* (2018).

[2] Martin Mundt et al. "A Wholistic View of Continual Learning with Deep Neural Networks: Forgotten Lessons and the Bridge to Active and Open World Learning". In: *ArXiv* abs/2009.01797 (2020).

[3] Robert M. French. "Using Semi-Distributed Representations to Overcome Catastrophic Forgetting in Connectionist Networks". In: 1991.

[4] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the National Academy of Sciences* 114 (2017), pp. 3521–3526.

[5] Noel E. Sharkey and Amanda J. C. Sharkey. "Backpropagation Discrimination Geometric Analysis Interference Memory Modelling Neural Nets". In: 1995.

[6] Anthony V. Robins. "Catastrophic forgetting in neural networks: the role of rehearsal mechanisms". In: *Proceedings 1993 The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems* (1993), pp. 65–68.

[7] Adrien Bardes, Jean Ponce, and Yann LeCun. "VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning". In: *ArXiv* abs/2105.04906 (2022).

[8] Mathilde Caron et al. "Unsupervised Learning of Visual Features by Contrasting Cluster Assignments". In: *ArXiv* abs/2006.09882 (2020).

[9] Mathilde Caron et al. "Emerging Properties in Self-Supervised Vision Transformers". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 9630–9640.

[10] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations". In: *ArXiv* abs/2002.05709 (2020).

[11] Xinlei Chen et al. "Improved Baselines with Momentum Contrastive Learning". In: *ArXiv* abs/2003.04297 (2020).

[12] Jure Zbontar et al. "Barlow Twins: Self-Supervised Learning via Redundancy Reduction". In: *ICML*. 2021.

[13] Zhenglin Zhu et al. "SWAV: a web-based visualization browser for sliding window analysis". In: *Scientific Reports* 10 (2020).

[14] Kai Tian, Shuigeng Zhou, and Jihong Guan. "DeepCluster: A General Clustering Framework Based on Deep Learning". In: *ECML/PKDD*. 2017.

[15] Debidatta Dwibedi et al. "With a Little Help from My Friends: Nearest-Neighbor Contrastive Learning of Visual Representations". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 9568–9577.

[16] Yonglong Tian, Olivier J Henaff, and Aäron van den Oord. "Divide and contrast: Self-supervised learning from uncurated data". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10063–10074.

[17] Gido M. van de Ven and Andreas Savas Tolias. "Three scenarios for continual learning". In: *ArXiv* abs/1904.07734 (2019).

[18] Matthias De Lange and Tinne Tuytelaars. "Continual Prototype Evolution: Learning Online from Non-Stationary Data Streams". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 8230–8239.

[19] Friedemann Zenke, Ben Poole, and Surya Ganguli. "Continual Learning Through Synaptic Intelligence". In: *Proceedings of machine learning research* 70 (2017), pp. 3987–3995.

[20] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. "Distilling the Knowledge in a Neural Network". In: *ArXiv* abs/1503.02531 (2015).

[21] Zhizhong Li and Derek Hoiem. "Learning without Forgetting". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40 (2018), pp. 2935–2947.

[22] Jonathan Schwarz et al. "Progress & Compress: A scalable framework for continual learning". In: *ArXiv* abs/1805.06370 (2018).

[23] Enrico Fini et al. "Self-Supervised Models are Continual Learners". In: *ArXiv* abs/2112.04215 (2021).

[24] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. "Growing a Brain: Fine-Tuning by Increasing Model

Capacity". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 3029–3038.

[25] Deboleena Roy, Priyadarshini Panda, and Kaushik Roy. "Tree-CNN: A hierarchical Deep Convolutional Neural Network for incremental learning". In: *Neural networks : the official journal of the International Neural Network Society* 121 (2020), pp. 148–160.

[26] Arun Mallya and Svetlana Lazebnik. "PackNet: Adding Multiple Tasks to a Single Network by Iterative Pruning". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 7765–7773.

[27] Joan Serrà et al. "Overcoming catastrophic forgetting with hard attention to the task". In: *ICML*. 2018.

[28] Lei Huang et al. "Normalization Techniques in Training DNNs: Methodology, Analysis and Application". In: *ArXiv* abs/2009.12836 (2020).

[29] Quang Pham, Chenghao Liu, and Steven Hoi. "Continual normalization: Rethinking batch normalization for online continual learning". In: *arXiv preprint arXiv:2203.16102* (2022).

[30] Sylvestre-Alvise Rebuffi et al. "iCaRL: Incremental Classifier and Representation Learning". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 5533–5542.

[31] David Lopez-Paz and Marc'Aurelio Ranzato. "Gradient Episodic Memory for Continual Learning". In: *NIPS*. 2017.

[32] Arslan Chaudhry et al. "Efficient Lifelong Learning with A-GEM". In: *ArXiv* abs/1812.00420 (2019).

[33] Ian J. Goodfellow et al. "Generative Adversarial Nets". In: *NIPS*. 2014.

[34] Chenshen Wu et al. "Memory Replay GANs: Learning to Generate New Categories without Forgetting". In: *NeurIPS*. 2018.

[35] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *CoRR* abs/1312.6114 (2014).

[36] Gido M. van de Ven, Hava T. Siegelmann, and Andreas Savas Tolias. "Brain-inspired replay for continual learning with artificial neural networks". In: *Nature Communications* 11 (2020).