ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10448256

# AUTONOMOUS GENERATIVE FEATURE REPLAY FOR NON-EXEMPLAR CLASS-INCREMENTAL LEARNING

*Yinjie Zhang* †, *Ming Shao* ⋆, *Wenlong Shi*†, *Haifeng Xia*†, *Siyu Xia*†

† School of Automation, Southeast University, China
⋆ Department of Computer and Information Science, University of Massachusetts Dartmouth, USA

## ABSTRACT

Deep neural networks have been successfully applied in many computer vision tasks. However, these models suffer catastrophic forgetting when learning new knowledge incrementally. To overcome the stability-plasticity dilemma, class incremental learning (CIL) has been widely discussed recently. The state-of-the-art CIL methods mainly leverage additional exemplar sets, thus memory costly and may raise privacy issues. To that end, we propose an autonomous generative feature replay (AGFR) framework without using exemplar sets. It consists of three modules: the feature extractor module, the feature generator module, and the unified classification module. First, to stabilize features over tasks, robust feature extractors are learned in a self-supervised manner and thus generalize well to unseen data. Second, instead of using exemplar sets or producing raw images, we propose an autonomous generative feature replay scheme to constantly update unified classifier in CIL without saving any image data. This strategy avoids overwhelming memory usage or poor quality of the generated raw images. Experiments demonstrate that our method achieves state-of-the-art performance in terms of average classification accuracy. ⋆

***Index Terms***— Knowledge reproduction, Generative representation learning, Incremental learning.

## 1. INTRODUCTION

Incremental learning has been discussed recently to enable deep neural networks to adapt to new tasks without complete re-training [1]. Despite great progress in incremental learning over the past few years, deep neural networks still suffer from "catastrophic forgetting" [2] when adapting to learn new knowledge. Although humans can continuously acquire new knowledge during their entire lives, learning systems must face a "stability-plasticity" dilemma and manage to consolidate knowledge in a progressive way [3].

Early research mainly focuses on task-incremental learning where prior such as task-ID is given at inference time. A more challenging setting termed "class-incremental learning (CIL)" becomes dominant where task-ID is no longer available [4]. Thus, a task-agnostic model is demanded to balance different classifier heads or learn a unified classifier.

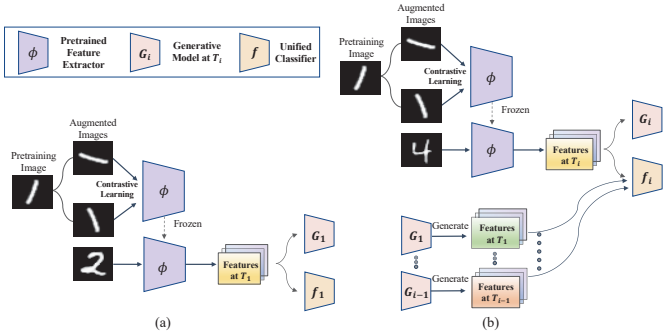


**Fig. 1**. Overview of our AGFR framework. (a) The model training process at the first incremental task to obtain $G_1$. (b) Model training process at the $i$-th incremental task $T_i$ ($i \geq 2$) to obtain $G_i$. Digits (e.g., 1, 2, 4) are used to illustrate input images of different classes. Feature extractor $\phi$ is trained at the beginning through SimCLR with several data augmentation tricks applied. Classifier $f$ is updated after each incremental task.

The overview of our work is shown in Figure 1. First, we design a robust feature extractor through self-supervised learning. This unsupervised learning fashion enables a generic representation of upcoming data, without needing any label information from them. When the incremental phases begin, we freeze the feature extractor and prevent it from being updated during the incremental training process. This benefits not only the generalization but also the computational efficiency. Second, image generation becomes more difficult with the increase in visual categories and diversities. Since CIL is meant to be a discriminant model, feature replay is more economical and affordable. Therefore, we propose a generative feature replay model. Compared to other methods, our feature generation strategy is simpler yet uses much less memory than exemplars-based CIL models. Third, we maintain a group of autonomous generative sub-models instead to address the issue of quality and diversity in image generation. The autonomous sub-models dedicated to each task are stored to avoid incremental training of GANs, and they jointly con-

⋆ This material is based upon work supported in part by the National Science Foundation under Grant No. 2144772.

tribute feature replay to CIL and unified classifier at new tasks. We found it very useful for alleviating performance degradation as the number of categories increases.

Our contributions could be summarized as the following three aspects. First, we propose to use self-supervised representation learning to pursue robust and generic features for existing and upcoming data and tasks. Second, a novel approach to replaying autonomous generative features without using an exemplar set is developed to offer a memory-efficient CIL solution. Finally, the results of the experiments demonstrate the superiority of our method compared to the state-of-the-art in terms of both accuracy and memory consumption.

## 2. PROPOSED ALGORITHM

### 2.1. Preliminaries and Motivation

Given a sequence of $m$ tasks $\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$ in incremental learning, CIL aims to learn and update the classifier $f$ over time to accommodate all the tasks and classes seen so far. Given a training dataset $\mathcal{D}_i = \{(x_j^i, y_j^i)\}_{j=1}^{n_i}$ at task $T_i$, where $(x, y)$ are image and label pairs and $n_i$ is the number of samples, the CIL algorithm aims to achieve the update: $\langle f_{i-1}, \mathcal{D}_i, \mathcal{E}\rangle \mapsto f_i$ where $f_i(x) \in \mathbb{R}^{c_i}$ is the softmax score vector, $c_i$ is the number of accumulated classes after the $i$-th task, and $\mathcal{E}$ is the optional exemplar set.

Most state-of-the-art CIL approaches store exemplars of previous tasks to rehearse the past knowledge [5, 6, 7, 8, 9]. Such strategies are not applicable to scenarios with data restriction (e.g., healthcare data), or limited memory (e.g., edge devices). The rest of CIL efforts have shifted to non-exemplar approaches and propose to use rehearsal approaches to replay or generate features/data of previous tasks [5, 10]. There are major issues regarding the quality and diversity of the data identified recently [11]. First, the visual quality of the generated images is difficult to guarantee, a common issue of many types of generative models such as GANs. Second, the diversity of data increases along with the upcoming tasks. It requires generative models to be incrementally updated and usually leads to poor training in practicals.

Our work considers the above drawbacks and improves the generative replay strategy. To address the first issue, we propose using self-supervised learning to extract robust features across different tasks and leverage the learned features for replay purposes. For the second issue, we propose a novel autonomous generative feature replay framework (AGFR). AGFR is essentially a "divide-and-conquer" approach: in the divide step, we split the entire generative model into several sub-models to avoid incremental training of a single generator; in the conquer step, we merge the classifiers from each task to offer a superior unified classifier for all tasks.

In our work, instead of raw images replay, we develop a robust feature extractor $\phi(x)$ as the intermediate representation to be fed to the autonomous generative models. In addition, we maintain a set of autonomous generative models $\mathcal{G} = \{G_1, G_2, ..., G_m\}$ for feature replay purpose. Here, $G_i$ is a generative model, e.g., GANs. Therefore, at task $T_i$, our AGFR algorithm aims to learn:

$$\langle f_{i-1}, \mathcal{D}_i, \phi, \bigcup_{j \le i-1} G_j \rangle \mapsto f_i, \tag{1}$$

where we do NOT use the exemplar set $\mathcal{E}$ when learning the new classifier $f_i$.

### 2.2. Autonomous Generative Feature Replay

In the first part, we introduce the design of our feature extractor module, which is trained in a self-supervised manner. In the second part, we explain the autonomous generative models and the replay mechanism, followed by the unified classification network to mitigate the catastrophic forgetting problem.

*2.2.1. Self-supervised Representation Learning*

The task-agnostic nature of CIL favors generalized representation, and a common latent space between different tasks would be preferred in this case. In this paper, we replace supervised feature learning with self-supervised learning methods. In particular, we adopt the popular SimCLR approach [12, 13] as our backbone, but other self-supervised learning frameworks could also be applied.

In summary, SimCLR learns representations by maximizing agreement between augmented views of the same data example via a contrastive loss in the latent space. We conduct stochastic data augmentation composed of three fundamental operations. First, we apply random cropping operations and then resize the augmented images back to the original size. Moreover, random color distortion operations and random Gaussian blur operations are used to produce two different views from one single image.

During the training process, we minimize contrastive loss. The loss function between a pair of examples $\{x_i, x_j\}$ augmented from the same input sample can be computed as:

$$\mathcal{L}_{SSL} = -\log \frac{\exp\Big(sim\big(\psi(z_i), \psi(z_j)\big)/\tau\Big)}{\sum_{k \neq i} \exp\Big(sim\big(\psi(z_i), \psi(z_k)\big)/\tau\Big)}, \tag{2}$$

where $\psi$ is a learnable embedding function, $sim(\cdot, \cdot)$ is cosine similarity between two vectors, and $\tau$ is a temperature parameter. Note the latent vector $z = \phi(x)$. The overall loss is computed across all positive pairs in a mini-batch. Once SimCLR training is complete, we freeze the model and use only $\phi(x)$ for feature extraction in CIL procedure.

One remaining question is what data shall be used for SimCLR training. Compared to the common strategy that incrementally updates feature extractors during the CIL process [5, 14], our feature extractor is trained only once before the CIL procedure without using any labeled data. To this end,

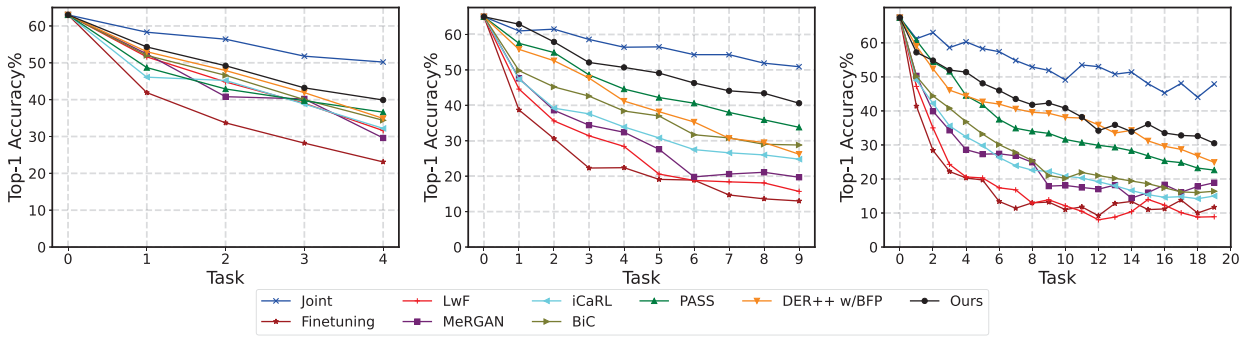


**Fig. 2**. Classification accuracy on CIFAR-100. The 100 classes of CIFAR-100 are equally divided into 5, 10, and 20 tasks.

we build a holdout unlabeled dataset without any intersection with the training or testing data of CIL. For instance, we divide TinyImageNet [15] into two subsets, each of which contains 100 classes. In experiments, one of them is used for pre-training, while the other is for training and evaluation. More details can be found in the experiment sections.

*2.2.2. Feature Replay*

This section details the learning of the set of autonomous generative model $\mathcal{G}$. At task $T_i$, the training process is divided into two steps. The first step is to learn $G_i$, and the second step is to learn an unified classification model $f_i$.

**Generative modeling.** For the first step, we adopt Wasserstein GANs with gradient penalty [16] to generate replayed features due to their promising ability reflecting the true distribution of complex data. Instead of adopting weight clipping as WGAN [17], penalizing the growth of gradient improves training stability and generation quality. In our autonomous generative replay framework, the generator $G$ and discriminator $D$ consists of three deconvolution and convolution layer, respectively. Here, we skip the subscript "$i$" for $\{G, D\}$ for simplicity. The input of the conditional generator $G$ includes a vector $v$ sampled from the Gaussian distribution and a corresponding label $y$. With a particular feature $z$, the improved objective function of the discriminator $D$ is:

$$\begin{aligned}\mathcal{L}_D = &\mathop{\mathbb{E}}_{v\sim\mathbb{N}(0,1)}\big[D(G(v,y)\big] - \mathop{\mathbb{E}}_{z\sim\mathbb{P}_r}\big[D(z)\big] \\ &+ \lambda \mathop{\mathbb{E}}_{\hat{z}\sim\mathbb{P}_{\hat{z}}}\big[(\|\nabla_{\hat{z}} D(\hat{z})\|_2 - 1)^2\big],\end{aligned} \tag{3}$$

where $\hat{z}$ is a random sample selected for gradient penalty, $z$ is a real sample, the latent vector $v$ is randomly sampled from Gaussian distribution accompanied by a given label $y$, and $\lambda$ is the balancing parameter. Additionally, $\mathbb{P}_r$ represents the real data distribution, and the distribution $\mathbb{P}_{\hat{z}}$ is sampled uniformly from straight lines which are drawn by pairs of points. One point of each pair is sampled from real data distribution, while the other is from generated data distribution.

**Unified classifier.** After the training of $G_i$, we will use the set of autonomous generative models learned so far, i.e., $\{G_1, ..., G_{i-1}\}$ to replay the features of previous tasks. Together with the feature extracted from task $T_i$, we are allowed to update the unified classifier $f_i$ which is a single-head network and can distinguish all seen classes from $T_1$ through $T_i$.

## 3. EXPERIMENTS

### 3.1. Experimental Setup

♢ **Datasets**. We evaluate the performance of our approaches and other state-of-the-art methods on CIFAR-100 [18] and TinyImageNet [15] datasets. In terms of the dataset partition, we follow the protocol of [19] and equally split the 100 classes of CIFAR-100 dataset into 5, 10 and 20 tasks prepared for incremental phases. Similarly, for TinyImageNet dataset, we first randomly select 100 classes of images from the entire dataset to pre-train the self-supervised feature extractor. Then the rest 100 classes are equally divided into 5, 10 and 20 tasks in the same way as the CIFAR-100 dataset.

♢ **Baselines & Evaluation Metrics**. We compare our method with three non-exemplar-based approaches: LwF [20], MeRGAN [21], PASS [22] and three exemplar-based methods iCaRL [5], BiC [23], BFP [19]. All exemplar-based methods have a limited memory size of 4,000. We also add Joint Training (JT) and Finetuning (FT) methods as the baseline to compare with our proposed AGFR framework. In particular, JT method is trained on all seen data together and can be treated as the upper-bound performance of incremental learning methods. FT method is considered as a naive baseline method for CIL since it trivially re-trains the classifier with the new data and possibly defines the lower-bound.

In this work, we employ the most critical evaluation metric: the average classification accuracy [22, 24]. This metric is defined as the average top-1 classification accuracy over all the incremental learning tasks. Phased classification accuracy $a_{i,j} \in [0,1](j \leq i)$ reflects the task $T_j$ performance after learning the current task $T_i$, which is computed after each incremental step. After task $T_k$, the overall classifier performance is evaluated by the average accuracy $\frac{1}{k}\Sigma_{j=1}^{k} a_{k,j}$.

♢ **Implementation Details**. In CIFAR-100 experiments, we pre-train the feature extractor framework using CIFAR-10 dataset [18] without labels. In TinyImageNet experiments, we pre-train the feature extractor with the presegmented subset. In all experiments, different from "Base50", which uses half of the categories of data for supervised CNN pre-training, we follow the "Base0" setting that conducts CIL from the first task [25]. This protocol allows us to evaluate our framework

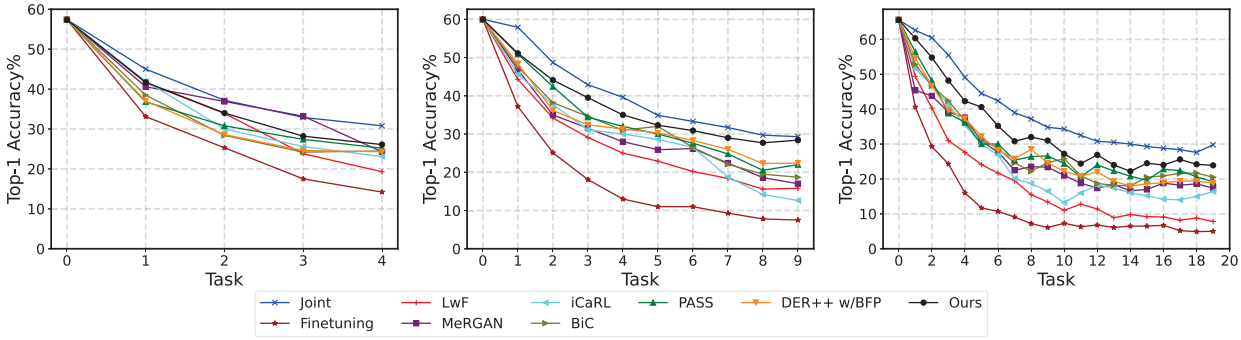


**Fig. 3**. Average classification accuracy on TinyImageNet. 100 classes of examples are randomly selected from the full TinyImageNet dataset and then equally divided into 5, 10 and 20 tasks.

| Methods | Datasets | Exemplar Number | Memory Size |
|---|---|---|---|
| Exemplar based | CIFAR-100 | 4000 Images | 18Mb |
| | TinyImageNet | 4000 Images | 70Mb |
| AGFR(Ours) | - | - | 12Mb(5 Tasks)<br>24Mb(10 Tasks)<br>48Mb(20 Tasks) |

**Table 1**. Comparison of memory cost across exemplar-based methods and our AGFR. The main memory overhead of our AGFR comes from the number of generative models, and is unrelated to image size and quantity.

with others under a more difficult setting on both datasets.

Our AFGR is implemented with PyTorch [26]. For both datasets, we adopt ResNet-50 [27] as the backbone of the self-supervised feature extractor and ResNet-18 as the backbone of the classifier. The feature extractor and the generative model are trained for 100 epochs, while the classifier is trained for 200 epochs, with Adam [28] as optimizer.

### 3.2. Experimental Results

For fair comparisons, all compared methods adopt ResNet-18 as the backbone to extract feature from raw images. Figure 2 and Figure 3 report the performance of the selected methods on Cifar-100 and TinyImageNet, respectively.

From Figure 2 and Figure 3, it is simple to find that our AGFR achieves better average accuracy than the existing methods. Especially, our AGFR framework surpasses PASS with the second best performance on most cases. Moreover, when comparing with non-exemplar-based algorithms ( e.g., LwF ), our method obtains significant improvement. The main reason lies in that our AGFR implicitly preserves data pattern of previous tasks into model and precisely reproduce high-quality samples for model adaptation on the novel tasks. Similarly, exemplar-based approaches also stores partial instances for incremental learning. However, our method explores "*feature replay*" mechanism to increase sample diversity and bring the additional benefits on avoiding knowledge forgetting, which results in the higher performance. These observations illustrate that our method continuously learns novel knowledge and effectively preserve semantic information of previous seen samples.

### 3.3. Analysis of Memory Consumption

In this section, we further discuss the required memory size of exemplar-based methods, including iCaRL, BiC, and DER++ w/BFP. In our experiment setting, all these methods have a fixed memory size of 4,000 images for all seen classes. Also, we compare the required memory consumption of our AGFR used for storing generative models with the above exemplar-based methods. The comparison result is shown in Table 1.

On the CIFAR-100 dataset, the memory consumption for the exemplar set is affordable due to the low resolution of raw images. However, the required memory space increases dramatically from CIFAR-100 to TinyImageNet dataset in order to host higher-resolution images in the exemplar set. With larger datasets and higher image resolution, the memory consumption will increase linearly in dataset size and quadratically in image size. Therefore, it becomes less reasonable to store more raw images to improve model performance. Note that although our approach requires more memory space on CIFAR-100 dataset under the setting of 10 and 20 divided tasks, it saves more space on the TinyImageNet dataset. Moreover, our AGFR framework is promising to be extremely space-saving on large-scale datasets with higher image resolution, such as ImageNet [29], compared to the state-of-the-art exemplar-based methods.

## 4. CONCLUSIONS

In this work, we proposed a novel generative feature replay framework to mitigate the catastrophic forgetting problem and data privacy issue in the setting of class incremental learning without using exemplar sets. Different from other class incremental approaches, our method proposed to use a self-supervised pre-trained feature extractor for better generalization in facing new tasks. Considering the fact that low-dimensional features are much easier to generate, we applied conditional GANs to replay features of previous tasks. We also used multiple autonomous generative models to deal with different tasks. Experiments illustrated that our method outperformed other state-of-the-art methods in average accuracy, especially when the number of tasks was large.

## 5. REFERENCES


[1] R. Polikar, J. Byorick, S. Krause, A. Marino, and M. Moreton, "Learn++: A classifier independent incremental learning algorithm for supervised neural networks," 2002. 1

[2] M. Mccloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *Psychology of Learning and Motivation*, vol. 24, pp. 109–165, 1989. 1

[3] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary environments: A survey," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 12–25, 2015. 1

[4] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer, "Class-incremental learning: survey and performance evaluation on image classification," 2022. 1

[5] S. A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," *IEEE Computer Society*, 2016. 2, 3

[6] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu, "Large scale incremental learning," *arXiv e-prints*, 2019. 2

[7] S. Hou, X. Pan, C. L. Chen, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," 2019. 2

[8] H. Ahn, J. Kwak, S. Lim, H. Bang, H. Kim, and T. Moon, "Ss-il: Separated softmax for incremental learning," 2021. 2

[9] H. Cha, J. Lee, and J. Shin, "Co$^2$l: Contrastive continual learning," 2021. 2

[10] Khurram Javed and Faisal Shafait, "Revisiting distillation and incremental classifier learning," 2018. 2

[11] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," 2018. 2

[12] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020. 2

[13] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," 2020. 2

[14] X. Liu, C. Wu, M. Menta, L. Herranz, B. Raducanu, A. D. Bagdanov, S. Jui, and Vdw Joost, "Generative feature replay for class-incremental learning," *IEEE*, 2020. 2

[15] Hadi Pouransari and Saman Ghili, "Tiny imagenet visual recognition challenge," 2014. 3

[16] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," 2017. 3

[17] Martin Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein gan," 2017. 3

[18] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Handbook of Systemic Autoimmune Diseases*, vol. 1, no. 4, 2009. 3

[19] Qiao Gu, Dongsub Shim, and Florian Shkurti, "Preserving linear separability in continual learning by backward feature projection," *arXiv preprint arXiv:2303.14595*, 2023. 3

[20] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE*, , no. 12, 2018. 3

[21] Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Van De Weijer Joost, and Bogdan Raducanu, "Memory replay gans: learning to generate images from new categories without forgetting," 2018. 3

[22] F. Zhu, X. Y. Zhang, C. Wang, F. Yin, and C. L. Liu, "Prototype augmentation and self-supervision for incremental learning," 2021. 3

[23] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu, "Large scale incremental learning," 2019. 3

[24] A. Chaudhry, P. K. Dokania, T. Ajanthan, and Phs Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," 2018. 3

[25] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu, "Deep class-incremental learning: A survey," *arXiv preprint arXiv:2302.03648*, 2023. 3

[26] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. Devito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017. 4

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2016. 4

[28] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Computer Science*, 2014. 4

[29] Jia Deng, Wei Dong, Richard Socher, Li Jia Li, Kai Li, and Fei Fei Dept, "Imagenet : A large-scale hierarchical image database," *Proc. CVPR, 2009*, 2009. 4