Task Ordering Matters for Incremental Learning

Zhaonan Yang
Shenzhen Institutes of Advanced Technology,
Chinese Academy of Sciences
University of Chinese Academy of Sciences
Shenzhen, China
zn.yang@siat.ac.cn

Huiyun Li
Shenzhen Institutes of Advanced Technology,
Chinese Academy of Sciences
Shenzhen, China
hy.li@siat.ac.cn

***Abstract*—Current incremental learning is confronted with the problem of catastrophic forgetting. Existing work usually addressed this problem by enlarging the sample database. But few people pay attention to the order-sensitive problem in incremental learning. In this article, we propose a task sorting method with the feature similarity between the consecutive tasks. Experimental results on CIFAR-100 and CORe50 datasets demonstrate that the learning sorting matters for incremental learning. The task sorted with the highest feature similarity will get a better performance than that of random sorting.**

***Index Terms*—incremental learning, feature similarity, task sorting method**

I. Introduction

Humans and animals have the ability to incrementally acquire and update knowledge throughout their lives. The ability to retain new knowledge while retaining previously learned knowledge is called *incremental learning or continual learning* [1], which is essential for building scalable and reusable artificial intelligence systems [2].

Current deep neural networks have achieved impressive performance on many benchmarks, comparable to or better than humans [3]. However, when being trained on a new task, these networks almost completely forget the previous task due to the *catastrophic forgetting* [1] between the new task and the previously learned tasks.

In recent years, many methods have been proposed to alleviate this problem that hinders the development of lifelong learning intelligence systems. The replay-based method performs better than other methods, especially in class-incremental learning [4]–[7]. Specifically, class-incremental learning defines the setting where the task-ID is unknown at inference time, making the predictions of the learned models task-agnostic, After learning, models must be able to both solve each task seen so far and infer which task they are presented with [4].

In addition to catastrophic forgetting, there is also another important but relatively less explored problem called order-sensitive problem [7] in incremental learning, which describes the performance discrepancy to the task arrival sequence. Task order-sensitive problem means that different task sorting will get different performance. Task sorting for class-incremental learning influence the overall evaluation performance. A proposed class sorting based on the confusion matrix can be used as a tool for checking the robustness of class-incremental learning approaches [8]. But this work only shows that different learning sortings will lead to different results and does not explain why and how to get a better sorting.

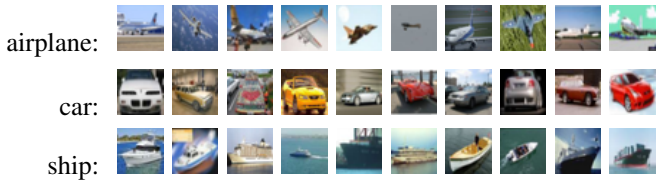


Fig. 1. **The CIFAR-100 dataset** consists of 60000 32x32 colour images in **100** classes, with 600 images per class. There are 50000 training images and 10000 test images. Above are the classes in the dataset, as well as 10 random images from each.

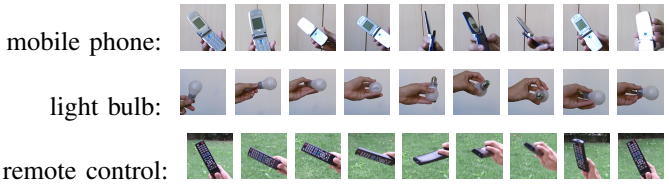


Fig. 2. **The CORe50 dataset** is a collection of 50 domestic objects belonging to 10 categories: plug adapters, mobile phones, scissors, light bulbs, cans, glasses, balls, markers, cups and remote controls. Classification can be performed at object level (**50** classes). Above are the classes in the dataset, as well as 9 random images from each.

In this paper, we propose the task sorting method based on cosine similarity to study the impact of feature similarity between adjacent tasks on catastrophic forgetting. Finally, we verify our algorithm on the CIFAR-100 dataset [9] as shown in Fig. 1. and the CORe50 dataset [10] as shown in Fig. 2.

The contributions of this paper are the following:

- Propose a task sorting method that combines feature extraction based on ResNet-18 [3], feature similarity based on cosine similarity, and task sorting algorithm based on the simulated annealing algorithm [11].
- Analyze the impact of feature similarity between adjacent tasks on catastrophic forgetting.
- Verify our proposed method on CIFAR-100 and CORe50.

The remainder of this paper is organized as follows. The related work is introduced in Section II. Section III presents the task sorting method proposed that combines Section III-A feature extraction, Section III-B feature similarity distance

978-0-7381-1316-6/21/$31.00 ©2021 IEEE

matrix, and Section III-C task sorting. Section IV details the experiments and results. Conclusion follows in Section V.

II. Related work

A. Methods for class-incremental learning

To overcome catastrophic forgetting several approaches, inspired in part by biological systems, have been proposed. These approaches can be divided into three series [12], [13] based on how task-specific information is stored and used throughout the sequential learning process [14], [15]. The first series is regularization-based methods, and this series of work [16]–[20] introduces additional regularization constraints in the loss function to consolidate previous knowledge while learning new data. The second series is parameter isolation methods, this series assigns different model [21]–[25] parameters to each task to prevent any possible forgetting. Some methods freeze the previous task parameters while adding new branches for new tasks to overcome catastrophic forgetting. The last one is replay methods, this series of work [26]–[32] stores samples in original format, or uses generative models to generate pseudo samples. Replay these previous task samples while learning new tasks to reduce forgetting. They can be reused as model inputs for exercises, or they can limit the optimization of new tasks to prevent interference from previous tasks.

In the recent large-scale comparative study of different methods, replay methods were found to produce far more reliable results than other approaches [4]–[7]. To directly extract feature data, we chose a replay method called generative feature replay [26] to achieve class-incremental learning and verified our conclusions.

B. Similarity distance

Humans learn new tasks based on similarity, the correlation between tasks affects human learning [33]. The information, similar classes in the dataset will be classified into a rough category, can be easily found in a dataset and the confusion matrix obtained through joint training also reflects the similarity between different classes.

To deeply analyze the impact of the similarity between different tasks on the incremental learning results, We extract feature information of each task and calculate the similarity between tasks using cosine similarity.

The cosine distance used to measure the similarity between two feature vectors $\mathbf{u_i}$ and $\mathbf{u_j}$ is

$$\begin{aligned} f\left(\mathbf{u}_i, \mathbf{u}_j\right) &= 1 - \frac{\mathbf{u}_i \mathbf{u}_j}{\|\mathbf{u}_i\| \|\mathbf{u}_j\|} \\ &= 1 - \frac{\sum_{d=1}^{D} \mathbf{u}_{id} \mathbf{u}_{jd}}{\sqrt{\sum_{d=1}^{D} {\mathbf{u}_{id}}^2} \sqrt{\sum_{d=1}^{D} {\mathbf{u}_{jd}}^2}} \end{aligned} \tag{1}$$

where D is the dimension of a feature vector and $\mathbf{u}_{id}$ is the value of the d-th dimension of $\mathbf{u}_i$.

C. Task sorting

Most papers use a random or a fixed class sorting for incremental learning training. There are only a handful of papers [13], [34] that consider the impact of learning order on catastrophic forgetting but only briefly described this phenomenon without special research on the specific task sequence. Use confusion matrix to get different class sorting [8]. A novel incremental learning model with additive parameter decomposition is proposed [35] to tackle the problem of order-sensitivity where the performance of the tasks largely varies based on the sorting of the task arrival sequence.

In this paper, we took the task as a unit to determine the task sorting by the feature similarity between adjacent tasks based on a feature similarity matrix rather than the relationship between classes based on a confusion matrix [8].

III. Task sorting method

A. Feature extraction

In the setting of incremental learning, task learning is based on dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathcal{X}$ is the i-th image, y_i (from a vocabulary of K classes) is the label of $\mathbf{x}_i$ and N is the number of all images in the dataset. We consider the incremental learning setting where T classification tasks are learned independently and in sequence from the corresponding datasets $D = \{D^1, ..., D^t, ..., D^T\}$, where $D^t \cap D^{t'} = \emptyset$ for all $t' \neq t$. The classes in each task are disjoint, i.e. $\mathcal{C}^t \cap \mathcal{C}^{t'} = \emptyset$ for all $t' \neq t$, where $\mathcal{C}^t$ is the corresponding label of dataset D_t.

To extract the feature data $\mathbf{u} = \{\mathbf{u}_i\}_{i=1}^{N}$ of the images $\mathbf{x} = \{\mathbf{x_i}\}_{i=1}^{N}$, the classifier network has the form, $\tilde{\mathbf{y}} = M(\mathbf{x}; \theta, V) = H(F(\mathbf{x}; \theta); V)$, where we explicitly distinguish between feature extractor $F(\mathbf{x}; \theta)$ parametrized by θ, and classifier $H(\mathbf{u}; V)$ parametrized by V, $\mathbf{u} = F(\mathbf{x}; \theta)$.

The resulting model M_t after learning task t has feature extractor F_t and classifier H_t. Ideally, after learning task t, the model can perform inference on all tasks $t' \leq t$, it remembers current and previous tasks.

B. Feature similarity matrix

A similarity function is a real-valued function that quantifies the similarity between two feature vectors. We used the average similarity between all feature vectors of two classes to measure the similarity between classes and used the average similarity between all classes of two tasks to measure the similarity between tasks.

The function $f(\mathbf{u}_i, \mathbf{u}_j)$ Eq. (1) is used to calculate the similarity between two feature vectors $\mathbf{u}_i$ and $\mathbf{u}_j$ ($\mathbf{u}_j$ is a feature vector). The formula for calculating the distance between the features of the i-th class and the j-th class is

$$distance\left(\mathbf{u}^i, \mathbf{u^j}\right) = \frac{1}{N_i N_j} \sum_{m=1}^{N_i} \sum_{n=1}^{N_j} f\left(\mathbf{u}_m^i, \mathbf{u}_n^j\right) \tag{2}$$

where N_i is the number of feature vectors in $\mathbf{u}^i$, and $\mathbf{u}_m^i$ is the m-th features vectors of $\mathbf{u}^i$.

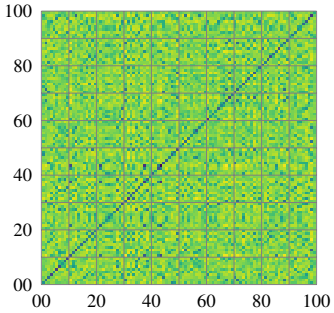


Fig. 3. A random feature similarity matrix under a random similarity task sorting. Both the horizontal and vertical axes represent the classes in CIFAR-100, and the value at any point in the figure represents the distance between the two classes. We use blue to indicate a lower distance meaning higher similarity, and yellow indicate the opposite.

After the similarity calculation of Eq. (2), a feature similarity matrix $M \in \mathbb{N}^{|K|\times|K|}$ of size $K * K$ can be obtained shown in Fig. 3, because there are K classes of images in total. Figs. 4(b) or 5(b) is also a similarity matrix.

C. Task sorting

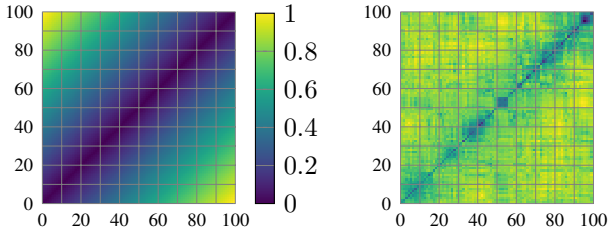
Incremental learning will split the total task into multiple tasks for learning during learning. The number of classes in a task will affect the results of incremental learning. According to the task division and simulated annealing algorithm, the sorting can be obtained.

1) Task division: We use the *step size* to represent the number of classes that need to be learned for each task in incremental learning. To determine which classes should be classified into a task, we used the confusion matrix learning all classes in a non-incremental way (usually known as joint training) and the coarse labels provided by the dataset to determine. Divide the total number of classes that require incremental learning (K') by step size to get the number of tasks (T) that need incremental learning after determining the value of step size, so $T = \frac{K'}{step\ size}$.

In this paper, the data in each experiment contains 100 classes and we set half of the classes as the first task and $step\ size$=2, so K=100, and K'=50, the number of tasks T corresponding to step size is T=25.

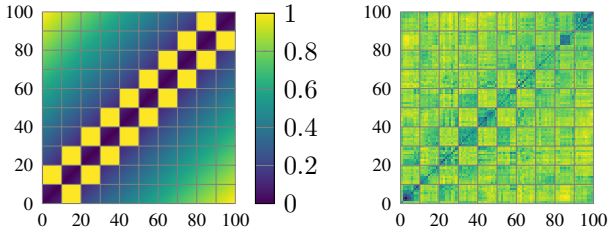
2) Simulated annealing algorithm: As a result, this paper analyzed the performance under different task sortings. To illustrate the effort of task sorting, we define three different task sortings according to feature similarity as:

- **Sorting tasks randomly (random-sorting)**: We use this sorting to represent others works's performance because most works on class-incremental learning report results by using a random order of classes on CIFAR-100 [36]–[38].
  The similarity between adjacent tasks has no correlation. A feature similarity between tasks is shown in Fig. 3. Both the horizontal and vertical axes represent the classes in CIFAR-100, and the value at any point in the figure represents the distance between the two classes and we use blue to indicate a lower distance meaning higher similarity, and yellow indicate the opposite.
- **Sorting tasks in ascending order (similar-sorting)**: The similarity between features of adjacent learning tasks is higher than that of non-adjacent tasks. Tasks with similar characteristics will be incrementally learned. A feature similarity matrix between tasks is shown in Fig. 4(b).
- **Sorting tasks in descending order (dissimilar-sorting)**: The similarity between features of adjacent learning tasks is lower than that of non-adjacent tasks. Tasks with large feature differences will be incrementally learned. A feature similarity between tasks is shown in Fig. 5(b).



(a) W_{high}: The desired high similarity matrix. (b) M_{high}: A high similarity matrix is found with a high similarity feature task sorting.

Fig. 4. W_{high} is the desired high similarity matrix for finding a high similarity matrix M_{high}. Task learning under this sorting can get a better performance than under a random.



(a) W_{low}: The desired low similarity matrix. (b) M_{low}: A low similarity matrix is found with a low similarity feature task sorting.

Fig. 5. W_{low} is the desired low similarity matrix for finding a low similarity matrix M_{low}. Task learning under this sorting can get a worse performance than under a random.

Finding the above similar or dissimilar task sorting based on the $K * K$ similarity matrix is a non-trivial task, where brute-force naive approach of O(n!) can not be applied.

To reduce the time complexity of the algorithm, we transform the problem of finding the sorting of tasks into an optimization problem where the purpose is to maximize the value of the function to a desired matrix W. In this case, we use the simulated annealing optimization algorithm to find sorting within the constraint time limit since it is difficult to establish a global value. Using this algorithm can quickly avoid local minimum and converge to a solution close to the global optimum.

In this paper, to find a high similarity task sorting like Fig. 4(b) M_{high}, we set the desired matrix to Fig. 4(a) W_{high}

TABLE I
SIMULATED ANNEALING ALGORITHM WE DESIGNED.

Algorithm 1 Simulated annealing algorithm we designed.

Input: s_0: A task sorting s_0.
$s \leftarrow s_0$ ▷ Initialize stat.
$e \leftarrow E(s_0)$ ▷ Initialize energy.
$T \leftarrow T_0$ ▷ Initialize temperature.
$k \leftarrow 1$ ▷ Initialize iteration times.
while $T > T_{min}$ and $k < k_{max}$ **do**
  for i in T_L **do**
    $s' \leftarrow neighbour(s)$ ▷ Select an adjacent state s' of state s.
    $\triangle E \leftarrow E(s') - E(s)$ ▷ Energy reduction.
    if $\triangle E \leq 0$ or $random(0,1) < exp(-\triangle E/(k*T))$ **then**
      $s \leftarrow s'$ ▷ Update sorting state.
      $e \leftarrow E(s')$ ▷ Update energy.
    end if
  end for
  $T \leftarrow T_{max}/(1+log(1+T))$ ▷ Update temperature.
  $k \leftarrow k+1$ ▷ Update iteration times.
end while
Output: s: The task sorting that makes the lowest energy $E(s)$.

TABLE II
$neighbor(s)$: FIND A ADJACENT SORTING STATE.

Algorithm 2 $neighbor(s)$: Find a adjacent sorting state.

Input: $s_0 \leftarrow s_{previous}$
$s_1 \leftarrow Swap(s_0)$ ▷ Swap the order of two tasks.
$s_2 \leftarrow Reversion(s_0)$ ▷ Reverse the order of tasks.
$s_3 \leftarrow Inserion(s_0)$ ▷ Choose a task and insert it elsewhere.
$s_{current} \leftarrow$ random select from $\{s_1, s_2, s_3\}$.
Output: $s_{current}$

that each row is a geometric sequence. In the same way, to find a low similarity task sorting like Fig. 5(b) M_{low}, we set the desired matrix to Fig. 5(a) W_{low} that each row is a geometric sequence.

In this paper, the optimal task sorting problem is converted to finding the state with the lowest energy $E(s)$.

$$E(s) = (1 - W) * M$$

The pseudo-code of the algorithm is as follows Table I.

A neighbor solution should be generated in Table I. To generate a adjacent sorting state of this solution, three adjacent procedures including $Swap$, $Reversion$, and $Insertion$ are existed [39]. The difference is that we only sort tasks instead of classes, and the order of classes in task is random. The pseudo-code of the adjacent solution is as follows Table II.

IV. EXPERIMENTS AND RESULTS

Dataset. In this paper, we used two datasets, CIFAR-100 and CIFAR-50&CORe50. We first evaluated the performance on CIFAR-100. The CIFAR-100 dataset consists of 60000 32x32 color images in 100 classes (K=100) with 600 images per class. There are 5000 training images and 1000 test images. CIFAR-100 images are padded with 4 pixels, from which 32×32 crops are randomly sampled. The original center crop is used for testing. Random horizontal flipping is used as data augmentation for dataset. Finally, we evaluated the performance based on CIFAR-50&CORe50 that also contains 100 classes, including half classes of CIFAR-100 and all classes of CORe50. All images are processed into the same size and the two datasets have the same number in a class.

Training. We used the ResNet-18 network with a 3×3 kernel in the first convolutional layer as a feature extraction network and trained the model from scratch. There are 500 training images per class, We used half of the classes as the first task and split the remaining classes (K'=50) into 25 tasks with equally distributed classes on three different task sortings. So each task needs to learn 2 ($step\ size$=2) new classes incrementally.

Evaluation. We selected the maximum probability across all heads as the chosen output and used average overall accuracy as an evaluation metric which is computed as the average accuracy of all tasks up to the current task. We compared the performance under three sortings proposed in Section III-C on CIFAR-100 whose performances show in Fig. 6(a) and CIFAR-50&CORe50 whose performances show in Fig. 6(b). The same experiments were repeated five times, and the average value was taken as the final performance.

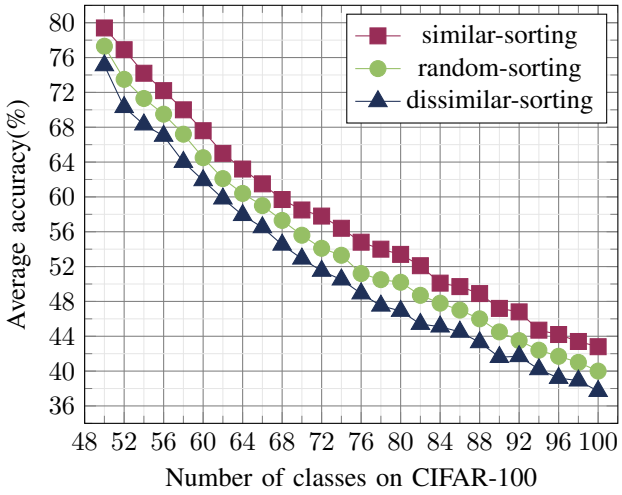
Experiments contain three different task sortings. In Fig. 6, the green solid line with circles represents the performance of incremental learning random tasks under the sorting with random tasks to learn adjacently (random-sorting), the red solid line with squares represents the performance of incremental learning high similar tasks under the sorting with similar tasks to learn adjacently (similar-sorting), the blue solid line with triangles represents the performance of incremental learning low similar tasks under the sorting with dissimilar tasks to learn adjacently (dissimilar-sorting).

The statistical result of average accuracy after leaning on 100 classes was listed in Table III after repeating each experiment five times.

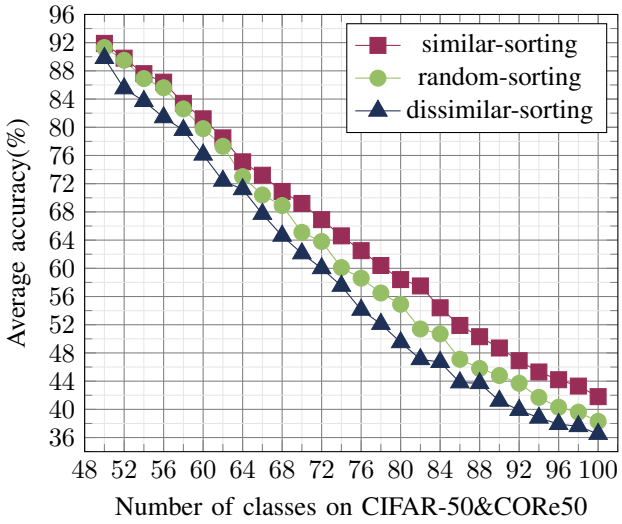
Results. After comparing three different colored lines in Fig. 6, we found that a better performance can be obtained through incremental learn high similar adjacent tasks under the sorting with similar tasks to learn adjacently, a worse performance can be obtained through incremental learning low similar adjacent tasks under the sorting with dissimilar tasks to learn adjacently, and an intermediate performance can be obtained through incremental learning random similar adjacent tasks under the sorting with random tasks to learn

TABLE III
THE AVERAGE ACCURACY (%) AFTER LEARNING 100 CLASSES ON DIFFERENT SORTINGS AFTER FIVE TIMES.

dataset	similar-sorting	random-sorting (other works's)	dissimilar-sorting
CIFAR-100	**57.48**±0.42	54.60±0.51	51.97±0.14
CIFAR-50& CORe50	**64.78**±0.71	61.87±0.56	58.48±0.31



(a) Incremental learning under three different task sortings (similar-sorting, random-sorting, and dissimilar-sorting) for **CIFAR-100** on ResNet-18. Performance best under similar-sorting.



(b) Incremental learning under three different task sortings (similar-sorting, random-sorting, and dissimilar-sorting) for **CIFAR-50&CORe50** on ResNet-18. Performance best under similar-sorting.

Fig. 6. Incremental learning under three different task sortings for CIFAR-100 and CIFAR-50&CORe50 datasets

adjacently. Performance best under similar-sorting. In general, incremental learning tasks with similar features are more conducive to overcoming catastrophic forgetting.

V. Conclusion

In this paper, we proposed a task sorting method that uses the cosine similarity between features to resorting the learning tasks. We analyzed the effect of different task sortings on class-incremental learning performance on CIFAR-100 and CORe50 datasets. We found that the best performance was obtained under the sorting with similar tasks to learn adjacently. This work is enlightening since the method is consistent with our biological practices, where learning with familiar but with slightly new knowledge will yield a good learning result.

Acknowledgment

This work was supported by CAS Key Laboratory of Human-Machine Intelligence-Synergy Systems, Shenzhen Institutes of Advanced Technology, and Shenzhen Engineering Laboratory for Autonomous Driving Technology, the Science and Technology Development Fun, Macao S.A.R. (FDCT) No.0015/2019/AKP.

References

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

[2] Z. Chen and B. Liu, "Lifelong machine learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 12, no. 3, pp. 1–207, 2018.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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

[5] J. von Oswald, C. Henning, J. Sacramento, and B. F. Grewe, "Continual learning with hypernetworks," *arXiv preprint arXiv:1906.00695*, 2019.

[6] G. M. van de Ven and A. S. Tolias, "Generative replay with feedback connections as a general strategy for continual learning," *arXiv preprint arXiv:1809.10635*, 2018.

[7] E. Belouadah and A. Popescu, "Il2m: Class incremental learning with dual memory," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 583–592, 2019.

[8] M. Masana, B. Twardowski, and J. van de Weijer, "On class orderings for incremental learning," *arXiv preprint arXiv:2007.02145*, 2020.

[9] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[10] V. Lomonaco and D. Maltoni, "Core50: a new dataset and benchmark for continuous object recognition," *arXiv preprint arXiv:1705.03550*, 2017.

[11] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated annealing: Theory and applications*, pp. 7–15, Springer, 1987.

[12] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[13] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *arXiv preprint arXiv:1909.08383*, 2019.

[14] S. Farquhar and Y. Gal, "Towards robust evaluations of continual learning," *arXiv preprint arXiv:1805.09733*, 2018.

[15] R. Aljundi, M. Lin, B. Goujaud, and Y. Bengio, "Online continual learning with no task boundaries," *arXiv preprint arXiv:1903.08671*, 2019.

[16] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," *Proceedings of machine learning research*, vol. 70, p. 3987, 2017.

[17] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 139–154, 2018.

[18] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[19] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 532–547, 2018.

[20] X. Liu, M. Masana, L. Herranz, J. Van de Weijer, A. M. Lopez, and A. D. Bagdanov, "Rotate your networks: Better weight consolidation and less catastrophic forgetting," in *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 2262–2268, IEEE, 2018.

[21] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7765–7773, 2018.

[22] C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, A. A. Rusu, A. Pritzel, and D. Wierstra, "Pathnet: Evolution channels gradient descent in super neural networks," *arXiv preprint arXiv:1701.08734*, 2017.

[23] J. Serra, D. Suris, M. Miron, and A. Karatzoglou, "Overcoming catastrophic forgetting with hard attention to the task," *arXiv preprint arXiv:1801.01423*, 2018.

[24] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[25] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3366–3375, 2017.

[26] X. Liu, C. Wu, M. Menta, L. Herranz, B. Raducanu, A. D. Bagdanov, S. Jui, and J. van de Weijer, "Generative feature replay for class-incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 226–227, 2020.

[27] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, "Experience replay for continual learning," in *Advances in Neural Information Processing Systems*, pp. 350–360, 2019.

[28] D. Isele and A. Cosgun, "Selective experience replay for lifelong learning," *arXiv preprint arXiv:1802.10269*, 2018.

[29] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Advances in neural information processing systems*, pp. 2990–2999, 2017.

[30] C. Atkinson, B. McCane, L. Szymanski, and A. Robins, "Pseudo-recursal: Solving the catastrophic forgetting problem in deep neural networks," *arXiv preprint arXiv:1802.03875*, 2018.

[31] F. Lavda, J. Ramapuram, M. Gregorova, and A. Kalousis, "Continual classification learning using generative models," *arXiv preprint arXiv:1810.10612*, 2018.

[32] J. Ramapuram, M. Gregorova, and A. Kalousis, "Lifelong generative modeling," *arXiv preprint arXiv:1705.09847*, 2017.

[33] B. Opitz and A. D. Friederici, "Brain correlates of language learning: the neuronal dissociation of rule-based versus similarity-based learning," *Journal of Neuroscience*, vol. 24, no. 39, pp. 8436–8440, 2004.

[34] M. Masana, T. Tuytelaars, and J. van de Weijer, "Ternary feature masks: continual learning without any forgetting," *arXiv preprint arXiv:2001.08714*, 2020.

[35] J. Yoon, S. Kim, E. Yang, and S. J. Hwang, "Scalable and order-robust continual learning with additive parameter decomposition," *arXiv preprint arXiv:1902.09432*, 2019.

[36] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

[37] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 831–839, 2019.

[38] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu, "Large scale incremental learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 374–382, 2019.

[39] M. Hajiaghaei-Keshteli and S. M. Sajadifar, "Deriving the cost function for a class of three-echelon inventory system with n-retailers and one-for-one ordering policy," *The International Journal of Advanced Manufacturing Technology*, vol. 50, no. 1-4, pp. 343–351, 2010.