Class Incremental learning for audio scene classifier using rehearsal-based strategy

Ibnu Daqiqil ID
岡山大学 阿部研究室
Abe Laboratory, Okayama University

Abstract— Novel class increamental is a special case of concept drift in which a new class comes into existence. To solve this problem, update or retrain the model is needed to accommodate new knowledge into a model. However, when the model is retrained using the latest dataset, the performance gradually decreases because the model forgets the previous knowledge. This situation is called catastrophic forgetting. Therefore, a straightforward solution is rehearsal training with all of the past dataset, but some systems have a limitation on the amounts of memory saved. This paper proposes a method for selecting the small portion of the past dataset as memory representative, and then we use GAN and data augmentation to generate the samples as an extension of the memory. Experimental results show promising results, prevent catastrophic forgetting, and increase backward transfer. The model performs better when using a low logit sample than a high logit in selecting the representative memory and GAN.

1 Background

The supervised machine learning scenario always works on the simple assumption that a full training dataset is available to model at once and always has the same distribution. However, in reality, data may come in batches, and its distribution might change. This phenomenon is known as concept drift. One of the concept drift cases is the novel class increment[1]. When a new class appear, we need to update the old model to accommodate the new class.

To solve this problem, incremental learning has emerged as the solution to adapt the models based on new data to accommodate new knowledge[2]. However, when we retrain the model to update the model using new data and update its parameters, the model only performs well on the new data. The parameter updates interfere with previously learned experience, leading to a drastic drop in performance on previously learned tasks—this phenomenon is known as catastrophic interference or forgetting [3].

Recent studies have shown promising results on rehearsal-based methods that use a small portion of previously learned data can retain learned experience and mitigate catastrophic forgetting [4], [5]. However, this approach can cause overfitting problems to the incoming data because the stored samples are much smaller than the incoming data, or the samples are ignored during training due to their small size. A straightforward solution is to increase the memory size as samples arrive gradually, but this does not preserve the important resource constraint of limited memory capacity in the problem settings. Therefore, we need a strategy that maintains the previously learned experience with only a small or limited sample.

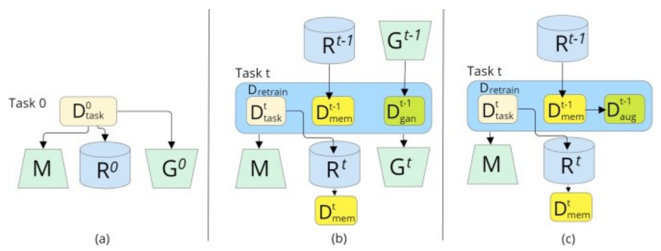


Fig. 1. Increamental training using rehearsal-based strategy. (a) In initial training, a model M is trained using current task dataset $\mathcal{D}^0_{task}$ and, a memory reservoir R and generator G^0 is builded to generate memory for next training. (b) In the incremental training using GAN, task dataset $\mathcal{D}^t_{task}$, past memory $\mathcal{D}^{t-1}_{mem}$ from memory reservoir and learned experience $\mathcal{D}^{t-1}_{gan}$ generated from previous generator $G^{t-1}$are used to retrain new M and G^t. (c) In incremental training using augmentation similar to GAN, the difference is that there is no generator and an augmented dataset $\mathcal{D}^{t-1}_{aug}$ augmented from $\mathcal{D}^{t-1}_{gan}$.

This chapter focuses on the rehearsal-based approach to maintaining information of the learned knowledge as much as possible with a few samples to solve class imbalance or overfitting problems. In [6], [7], we find that samples produced by the generative method are accurate enough to retain knowledge. However, the algorithm's efficacy heavily depends on the quality of the generator. So, our approach stores one generator to generate representatives of all the previous samples. Then we use the generator to generate samples and retrain the model with memory to avoid bias in the model.

2 Rehearsal-Based Incremental Learning

In this chapter, we propose a rehearsal framework that allows training deep convolutional networks in classification using representative memory and pseudo-memory.The framework is based on the learning strategy known as Rehearsal. In the real world, the teaching-learning process consists of the literal repetition of the exact words that students are required to remember, in oral or written form, using simple repetition, cumulative repetition, taking notes, and the marking or highlighting of texts[13]. In the context of computational neural networks, the rehearsal process can be modelled by storing data from previous training and then reusing that data to strengthen the learnt network and accommodate new knowledge into the network[14].

We consider training and evaluating our model over a sequence of tasks $\mathcal{T} = (\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_N)$ of N tasks. Each of task $\mathcal{T}_t$ has dataset $\mathcal{D}^t_{task} = \{x^t_n, y^t_n\}^{N_t}_{n=1}$ that contain N_t datapoints and labels. In $\mathcal{T}_t$, we assume that y^t has unique classes, $y^t \cap \{y^0 .. y^{t-1}\} = \emptyset$. Our proposed method consists of three modules:

A. Classifier

In the experiment we use a convolutional neural network (CNN) based model to classify a audio scene. The CNN consists of several convolutional layers where each convolutional layer contains several kernels that are convolved with the input feature maps to capture their local patterns. A classifier M consists of feature extractor F and classifier head C, $M(F(x;\theta);C)$. To classify data x, M extract feature u using F that parameterise by θ. C needs V as a matrix that projects u to class score using $\mathcal{A}$ softmax function, $C = \mathcal{A}(Vu)$, to classify the feature. The F itself contains 12 layers of CNN with Batch Normalisation and Dropout layer. Detail network architecture showed in Fig.2.

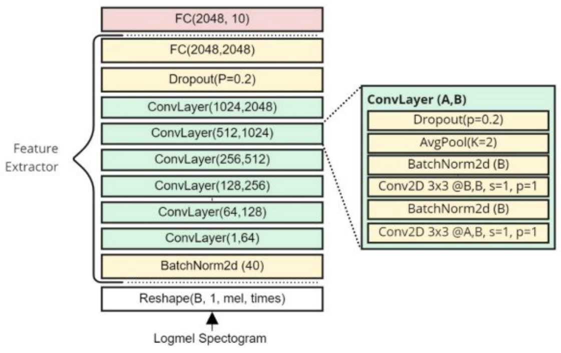


Fig. 2. Classifer Architecture

B. Rehearsal Memory

Rehearsal memory is a subset of representative samples of the previous dataset, $\mathcal{D}_{mem} \subset \mathcal{D}_{task}$. When the M is trained, the rehearsal memory $\mathcal{D}^t_{mem}$ is selected from $\mathcal{D}^t_{task}$ and they are stored in the memory reservoir R^t. $\mathcal{D}^t_{mem}$ was selected using several methods:

- **High or low probability**. This method uses the number generated from $\mathcal{A}$ that indicates the level of probability of the classification result. The higher the value, the more likely the classifier is to be correctly classified. Using a high logit means we focus on storing learning data with a high probability of being classified correctly. Conversely, if we choose a low probability, we store the representative memory containing data that is likely to be misclassified.
- **Mean clustering**. Mean clustering utilizes the average feature u to select data to be stored in representative memory. The smaller the distance from the average means that the data selected is data that frequently appears in the dataset.
- **Barycenter**. The concept used in this method is similar to using mean clustering. However, the samples selected are samples whose u are the closest to their moving barycenter distance [15].
- **Random selections**. Randomly selected samples from the current dataset.

C. Pseudo-rehearsal Memory

Pseudo-rehearsal memory is a set of generated samples that used in the retraining process. In this chapter, pseudo-rehearsal memory can be generated in two ways: using the generative method and data augmentation. In the generative method, we train our generator G^t using Memory Replay GAN (MER-GAN)[7] to generate dataset $\mathcal{D}^t_{gan}$. Generative adversarial networks (GANs) are a popular framework for image generation due totheir capability to learn a mapping between a low-dimensional latent space and a complex distribution of interest, such as natural images. A GAN consists of two networks, a Generator and a Discriminator, competing with each other in a zero-sum game framework.

MER-GAN consists of three components: generator, discriminator and classifier. The discriminator and classifier share all layers but the last ones (task-specific layers). The conditional generator is parametrized by θ^G and generate a sample $\tilde{x} = G_{\theta^G}(z,c)$ given a latent vector z and a class c. Similarly, the discriminator parametrized by θ^D tries to discern whether an input x is real or generated, while the generator tries to fool it by generating a more realistic sample. In addition, MER-GAN uses an auxiliary classifier C with parameter θ^C to predict the label $\tilde{c} = C_{\theta^C}(x)$, and thus forcing the generator to generate images that can be classified in the same way as real images

To train MER-GAN, we use join-retraining with replayed samples. The generator has an active role by replaying memories of previous tasks (via generative sampling) and using them during the training of the current task to prevent forgetting. Firstly we create dataset $\mathcal{D}^t_{gan}$ contain generated sample from all previous tasks from task 0 to t. To update M and G^t, we use the retrain dataset $\mathcal{D}^t_{retrain}$, which contain $\mathcal{D}^t_{task}$, $\mathcal{D}^{t-1}_{mem}$ and $\mathcal{D}^{t-1}_{gan}$. The illustration of the retraining procedure with GAN can be found in Fig. 1(b).

Once the extended dataset is created, the network is trained using joint training as

$$\min_{\theta^G_t}\left(L^G_{GAN}(\theta_t, \mathcal{D}^t_{retrain}) + \lambda_{CLS} L^G_{CLS}(\theta_t, \mathcal{D}^t_{retrain})\right) \tag{1}$$

$$\min_{\theta^D_t}(L^D_{GAN}(\theta_t, \mathcal{D}^t_{retrain}) + \lambda_{CLS} L^D_{CLS}(\theta_t, \mathcal{D}^t_{retrain})) \tag{2}$$

$$L^G_{GAN}(\theta_t, \mathcal{D}^t_{retrain}) = -\mathbb{E}_{z\sim p_z, c\sim p_c}[D_{\theta^D}(G_{\theta^G}(z,c))] \tag{3}$$

$$L^G_{CLS}(\theta_t, \mathcal{D}^t_{retrain}) = -\mathbb{E}_{z\sim p_z, c\sim p_c}\left[y_c log\ C_{\theta^C}\left(G_{\theta^G}(z,c)\right)\right] \tag{4}$$

$$L^D_{GAN}(\theta_t, \mathcal{D}^t_{retrain}) = -\mathbb{E}_{(x,c)\sim S}[D_{\theta^D}(x)] + \mathbb{E}_{z\sim p_z, c\sim p_c}[D_{\theta^D}(G_{\theta^G}(z,c))] + \lambda_{GP}\mathbb{E}_{x\sim S, z\sim p_z, c\sim p_c, \epsilon\sim p_\epsilon}\left[\left(\left\|\nabla D_{\theta^D}(\epsilon x + (1-\epsilon)G_{\theta^G}(z,c))\right\| - 1\right)^2\right] \tag{5}$$

$$L^D_{CLS}(\theta_t, \mathcal{D}^t_{retrain}) = -\mathbb{E}_{(x,c)\sim S}[\ C_{\theta^C}(G_{\theta^G}(z,c))] \tag{6}$$

The GAN loss uses the WGAN formulation with gradient penalty where $L^G_{GAN}(\theta_t, \mathcal{D}^t_{retrain})$ and $L^G_{CLS}(\theta_t, \mathcal{D}^t_{retrain})$ are loss for generator and the cross entropy loss for classification, respectively, $p_c = U(1,t)$, $p_z = N(0,1)$ are the sampling distributions (uniform and Gaussian, respectively), y_c is the one-hot encoding of c for computing the cross-entropy, ϵ are parameters of the gradient penalty term, sampled as $p_\epsilon = U(0,1)$ and the last term of L^D_{GAN} is the gradient penalty.

In the augmentation method, the dataset $\mathcal{D}^{t}_{retrain}$contain $\mathcal{D}^{t}_{task}$, $\mathcal{D}^{t-1}_{mem}$ and $\mathcal{D}^{t-1}_{aug}$. $\mathcal{D}^{t-1}_{aug}$ is the augmentation result of $\mathcal{D}^{t-1}_{mem}$ using Frequency masking[16]. This method masks several blocks of consecutive frequency channels by a uniform distribution. The illustration of the retraining procedure can be found in Fig. 1(c).

3 Experiment Setup

We use the TAU Urban Acoustic Scenes 2019. The dataset has 10 classes divided into 5 tasks, where each task contains two unique classes. In the experiment, we compared the use of Pseudo-memory with GAN alone with low and high representative memory $\mathcal{D}_{mem}$. Table 1 shows the detailed experiment scenario.

Table 1. Experiment Scenario

Experiment Scenario	$\mathcal{D}_{mem}$	$\mathcal{D}_{gan}$	$\mathcal{D}_{aug}$
GAN Alone (S0)	-	100%	-
Low Memory GAN (S1)	10%	90%	-
Low Memory Augmented (S2)	-	90%	10%
High Memory GAN (S3)	75%	25%	-
High Memory Augmented (S4)	-	25%	75%

To measure the proposed method, we use average accuracy and Backward Transfer (BWT). BWT measures the influence that learning a task has on the performance of previous tasks.

$$\mathrm{ACC} = \frac{1}{T}\sum_{i=1}^{T} \mathrm{Acc}_{T,i} \tag{7}$$

$$\mathrm{BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1} Acc_{T,i} - Acc_{i,i} \tag{8}$$

where T is the number of tasks and $Accuracy_{i,i}$ is the test accuracy score for task j after the model learned task i. We train our classifier task for 100 epochs and GANs for 500 epochs. The Adam optimizer is used in all experiments, and the learning rate for classifier dan GANs are 1e-4 and 1e-4, respectively.

4 Result and Discussion

Using GAN alone or S0 to regenerate samples from prior knowledge shows that this strategy can overcome catastrophic forgetting, but the model's performance degrades over time. The overall average accuracy using S0 is 0.733. Fig. 3 shows that the accuracy of S0 decreases as the number of classes increases. Furthermore, according to the detailed analysis, the task accuracy results show no positive backward transfer in retraining. Positive backward transfer is the influence of the current training that improves the accuracy of the previous tasks. So this method (S0) is not recommended in high incremental phases.

The use of $\mathcal{D}_{mem}$ both with $\mathcal{D}_{gan}$ or $\mathcal{D}_{aug}$ can be used to overcome the problem of data degradation. $\mathcal{D}_{aug}$ show promising results at S4, especially for the cluster mean method with an average accuracy of 0.7735. The cluster mean method also showed the highest accuracy for this combination at high and low memory counts. However, at S2, the overall accuracy is lower than S0 in most memory selection methods.

The experimental results show that the use of $\mathcal{D}_{gan}$ outperforms $\mathcal{D}_{aug}$ both in high and low memory. When using this combination,

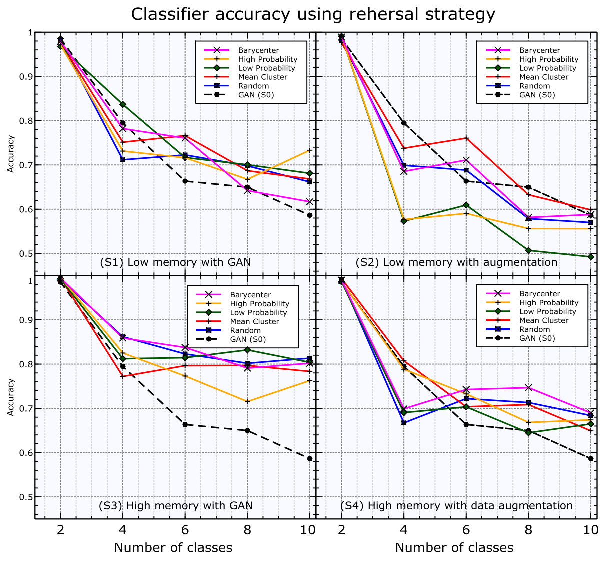


Fig. 3. Model accuracy in high and low representative memory

the classifier's performance improved and showed more stable accuracy, and some backward transfer was found, both in high and low memory. The average accuracy of the GAN combination in S1 and S3 were 0.7651 and 0.8418, respectively.

In the S1, using the random selection method shows the best results with an overall accuracy of 0.8581. However, selecting samples with low logit values shows the best performance in S3 with an average accuracy of 0.7805. The use of low logit also shows a higher average backward transfer than high probability, with an overall accuracy of 0.8510.

In this experiment, the larger number of representative memories is the larger positive BWT. The largest positive BWT was obtained when using high probability feature on low and high representative memory with GAN (S1 and S3) of 0.0222 and 0.0041, respectively. Detail accuracy of these scenario can be seet at Fig.4 and Fig. 5. In Fig.4, in task 1 until 5 there are many accuracy improvement after learning new task.

Table 1. Positive BWT result

	Representative Memory Selection				
	High Prob.	Low Prob.	Random	Bary Center	Cluster
S1	**0.0041**	0	0	0	0
S2	0	0	0	**0.0116**	0
S3	**0.0223**	0.0037	0.0055	0.0141	0.0034
S4	0.0134	**0.0303**	0.0085	0.0470	0.0399

We also compare the memory usage of the original dataset, representative memory, and GAN model size. The original dataset stores 3.9 GB of audio per class. For high and low $\mathcal{D}_{mem}$ require 73.6 MB and 16 MB per class. Our generator G requires a constant memory requirement of 82.85 MB for all classes. So by using a generator, the size of memory used remains the same, even if the number of experience models increases over time.

5 Conclusion

In this paper, we propose a framework to train audio scene classifiers using a rehearsal-based strategy incrementally. This method consists of three modules: classifier, rehearsal memory and pseudo-rehearsal memory. Rehearsal memory obtained by selecting the small portion of the past dataset as memory representative. Then we use GAN to generate the samples as an extension of the representative memory to avoid overfitting. Experimental results show promising results, prevent catastrophic forgetting, and increase backward transfer. The model performs better when using a low logit sample rather than a high logit in selecting the representative memory.

6 References

[1] M. Masud, J. Gao, L. Khan, J. Han, and B. M. Thuraisingham, "Classification and Novel Class Detection in Concept-Drifting Data Streams under Time Constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 6, pp. 859–874, Jun. 2011, doi: 10.1109/TKDE.2010.61.

[2] B. Gepperth, Alexander Hammer, "Incremental learning algorithms and applications," 2016, [Online]. Available: https://hal.archives-ouvertes.fr/hal-01418129.

[3] M. McCloskey and N. J. Cohen, "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem," *Psychol. Learn. Motiv.*, vol. 24, pp. 109–165, 1989, doi: 10.1016/S0079-7421(08)60536-8.

[4] J. Yoon, D. Madaan, E. Yang, and S. J. Hwang, "Online Coreset Selection for Rehearsal-based Continual Learning," in *International Conference on Learning Representations (ICLR)*, 2021, pp. 1–16.

[5] A. Prabhu, P. H. S. Torr, and P. K. Dokania, "GDumb: A Simple Approach that Questions Our Progress in Continual Learning," in *ECCV 2020*, 2020, pp. 524–540.

[6] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual Learning with Deep Generative Replay," in *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, May 2017, pp. 2994–3003, [Online]. Available: http://arxiv.org/abs/1705.08690.

[7] C. Wu, L. Herranz, X. Liu, Y. Wang, J. van de Weijer, and B. Raducanu, "Memory Replay GANs: learning to generate images from new categories without forgetting," in *32nd International Conference on Neural Information Processing*, Sep. 2018, pp. 1–12.

[8] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual Lifelong Learning with Neural Networks: A Review," Feb. 2018, doi: 10.1016/j.neunet.2019.01.012.

[9] R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, "Embracing Change: Continual Learning in Deep Neural Networks," *Trends Cogn. Sci.*, vol. 24, no. 12, pp. 1028–1040, Dec. 2020, doi: 10.1016/j.tics.2020.09.004.

[10] R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, "Embracing Change : Continual Learning in Deep Neural Networks," *Trends Cogn. Sci.*, vol. 24, no. 12, pp. 1028–1040, 2020, doi: 10.1016/j.tics.2020.09.004.

[11] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 6468–6477, 2017.

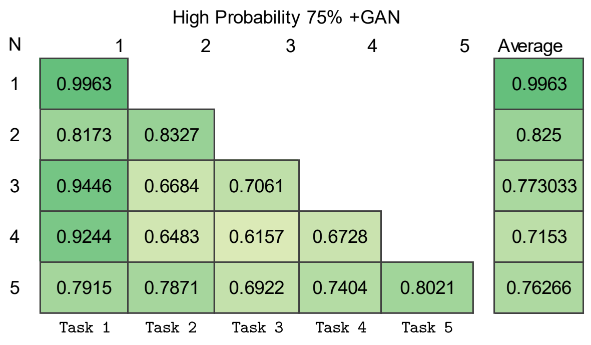
[12] J. Bang, H. Kim, Y. J. Yoo, J. W. Ha, and J. Choi, "Rainbow Memory: Continual Learning with a Memory of Diverse Samples," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8214–8223, doi: 10.1109/CVPR46437.2021.00812.

[13] L. M. Joseph, "Incremental Rehearsal: A Flashcard Drill Technique for Increasing Retention of Reading Words," *Read. Teach.*, vol. 59, no. 8, pp. 803–807, May 2006, doi: 10.1598/RT.59.8.8.

[14] D. Muñoz, C. Narváez, C. Cobos, M. Mendoza, and F. Herrera, "Incremental learning model inspired in Rehearsal for deep convolutional networks," *Knowledge-Based Syst.*, vol. 208, p. 106460, Nov. 2020, doi: 10.1016/j.knosys.2020.106460.

[15] S. A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, pp. 5533–5542, doi: 10.1109/CVPR.2017.587.

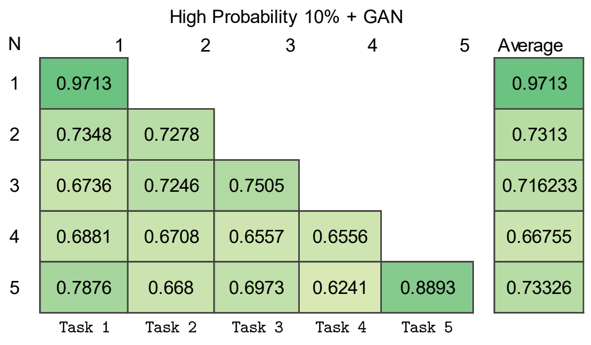
[16] D. S. Park *et al.*, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Interspeech 2019*, Sep. 2019, pp. 2613–2617, doi: 10.21437/Interspeech.2019-2680.



High Probability 75% +GAN

N	1	2	3	4	5	Average
1	0.9963					0.9963
2	0.8173	0.8327				0.825
3	0.9446	0.6684	0.7061			0.773033
4	0.9244	0.6483	0.6157	0.6728		0.7153
5	0.7915	0.7871	0.6922	0.7404	0.8021	0.76266
	Task 1	Task 2	Task 3	Task 4	Task 5	

Fig. 4. Model accuracy in high and low representative memory



High Probability 10% + GAN

N	1	2	3	4	5	Average
1	0.9713					0.9713
2	0.7348	0.7278				0.7313
3	0.6736	0.7246	0.7505			0.716233
4	0.6881	0.6708	0.6557	0.6556		0.66755
5	0.7876	0.668	0.6973	0.6241	0.8893	0.73326
	Task 1	Task 2	Task 3	Task 4	Task 5	

Fig. 5. Model accuracy in high and low representative memory