CONTINUAL LEARNING FOR ON-DEVICE ENVIRONMENTAL SOUND CLASSIFICATION

Yang Xiao[1,*], *Xubo Liu*[2,*], *James King*[2], *Arshdeep Singh*[2], *Eng Siong Chng*[1],
Mark D. Plumbley[2], *Wenwu Wang*[2]

[1] School of Computer Science and Engineering, Nanyang Technological University, Singapore
[2] Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey, UK

ABSTRACT

Continuously learning new classes without catastrophic forgetting is a challenging problem for on-device environmental sound classification given the restrictions on computation resources (e.g., model size, running memory). To address this issue, we propose a simple and efficient continual learning method. Our method selects the historical data for the training by measuring the per-sample classification uncertainty. Specifically, we measure the uncertainty by observing how the classification probability of data fluctuates against the parallel perturbations added to the classifier embedding. In this way, the computation cost can be significantly reduced compared with adding perturbation to the raw data. Experimental results on the DCASE 2019 Task 1 and ESC-50 dataset show that our proposed method outperforms baseline continual learning methods on classification accuracy and computational efficiency, indicating our method can efficiently and incrementally learn new classes without the catastrophic forgetting problem for on-device environmental sound classification.

Index Terms— Continual learning, environmental sound classification, on-device, convolutional neural networks

1. INTRODUCTION

Environmental sound classification aims to categorize audio recordings into pre-defined environmental sound classes [1]. Recently, on-device environmental sound classification [2, 3, 4] has attracted increasing research interest, as shown in Task 1 of Detection and Classification of Acoustic Scenes and Events (DCASE) 2022 Challenge: "Low-Complexity Acoustic Scene Classification" [5]. Such a sound classification system with low computation-complexity can be deployed on mobile and embedded platform for many real-world audio applications, such as acoustic surveillance [6], bio-acoustic monitoring [7] and multimedia indexing [8].

Most existing environment sound classification models [1, 3, 4, 9, 10] are trained with limited sound classes, which cannot directly adapt to new sound classes. When model developers want to expand the categories of environmental sounds to be classified, one way to do this is to fine-tune the model with new classes of data [11, 12]. However, this method may discard previously learned knowledge during the fine-tuning process: this is also known as the catastrophic forgetting problem [13]. Another possible solution is to re-train sound classification models with a mixture of historical and new data. However, this method is resource- and time-consuming in real-world on-device scenarios. As the solution based on re-training is computationally expensive, it is important to design efficient and effective methods to adapt the trained on-device sound classification model to new sound classes.

Continual learning (CL) [14, 15, 16] aims to continuously learn new knowledge over time while retaining and reusing previously learned knowledge. Existing CL methods can be generally divided into two categories: regularization-based methods [17, 18] and replay-based methods [19, 20]. Regularization-based methods use a regularization loss to preserve previously learned model parameters when learning new knowledge. Replay-based methods use a memory update algorithm (MUA) [20, 21, 22] to sample a few informative examples from historical data. The selected examples are used to preserve information about old classes when training new classes. Recently, replay-based CL methods have shown promising results outperforming regularization-based methods in audio tasks such as keywords spotting [23, 24] and sound event detection [25]. However, CL in on-device applications, such as on-device environmental sound classification, has received less attention in the literature, which is the focus in this paper. The on-device scenarios are often associated with restrictions in storage and memory space [3], which can pose challenges to replay-based CL which relied on external memory to restore historical data. As a result, the sound classification models that can be operated on the device may be limited in their capacities, thus prone to forgetting old knowledge when continuously learning new sound classes.

In this work, we investigate the replay-based CL (RCL) methods for on-device environmental sound classification. We first study the performance of existing memory update algorithm (MUA) methods such as *Reservoir* [21], *Prototype* [20] and *Uncertainty* [22] (as described in Section 2.1) on RCL for on-device environmental sound classification. We empirically demonstrate that *Uncertainty* [22] method performs best in our scenario. Furthermore, we propose *Uncertainty*++, a simple yet efficient MUA method based on *Uncertainty* method. Different to the *Uncertainty* method, our proposed *Uncertainty*++ introduces the perturbations to the embedding layer of the classifier. As a result, the computation cost (e.g., running memory and time) can be significantly reduced when measuring the data uncertainty. We evaluate the performance of our method on the DCASE 2019 Task1 [26] and the ESC-50 [27] datasets with on-device model BC-ResNet-Mod (∼86k parameters) [28, 29]. Experimental results show that *uncertainty*++ outperforms the existing MUA methods on classification accuracy, indicating its potential in real-world on-device audio applications. Our proposed method is model-independent and simple to apply. Our code is made available at the GitHub[1].

The remainder of this paper is organized as follows. Section 2 introduces the continual learning method we proposed for on-device environmental sound classification. Section 3 and Section 4 present the experimental settings and the evaluation results. Conclusions and future directions are given in Section 5.

[*] The first two authors contributed equally to this work.

[1] `https://github.com/swagshaw/ASC-CL`

2. METHOD

This section first describes replay-based continual learning and four memory update algorithms, and then introduces the proposed *uncertainty++* algorithm.

2.1. Replay-based continual learning

Following the continual learning setting [14, 18, 25] of environmental sound classification, we assume that the model M should identify all classes in a series of tasks $T = \{\tau_0, \ldots, \tau_t\}$ without catastrophic forgetting. For each task $\tau \in T$, we have input pairs (x, y) and classes C, where x denotes audio waveforms and y are classes $c \in C$. We aim to minimize a cross-entropy loss of all classes C present in the current task τ formulated as:

$$L_{CE}(\tau) = \sum_{c \in C} y_c log \frac{exp(M(x)_c)}{\sum\limits_{c \in C} exp(M(x)_c)}, \tag{1}$$

Where $M(x)$ denotes the output of the model M for input x.

The parameters learned from the previous task are potentially overwritten after learning the new class, also known as catastrophic forgetting. To mitigate this issue, we introduce replay-based methods. The replay-based methods utilize a region of the memory which is called 'replay buffer' to temporarily store the historical training samples to maintain the performance.

Re-training sound classification models with the mixture of the whole historical and new data is resource- and time-consuming in real-world on-device scenarios. To mitigate this issue, the replay-based methods access only a subset of the historical data to save the storage space. In this case, how to select the part of samples to the replay buffer by the memory update algorithm is the key.

Specifically, in the training of task τ_t, the replay buffer stores the selected training samples from the previous $t-1$ learned task(s) $\{\tau_0, \tau_1, \ldots, \tau_{t-1}\}$, and builds the training data buffer $\hat{D}_t$ for task τ_t formulated as:

$$\hat{D}_t = g(\hat{D}_{t-1}) \cup D_t, \tag{2}$$

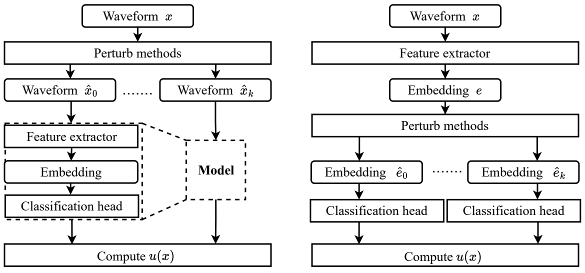
where g is the memory update algorithm [24], $\hat{D}_{t-1}$ is the training data buffer for task τ_{t-1}, and D_t is the incoming data for the new task.

2.1.1. Memory update algorithm (MUA)

We introduce four memory update algorithms in the literature. Generally, we assume that the memory update should select L samples from the training data $\hat{D}_{t-1}$ of the previous task τ_{t-1} for the training of the task τ_t.

Random [30] memory update algorithm selects L new samples $\{(x_1, y_1), (x_2, y_2), \ldots, (x_L, y_L)\}$ for the next task randomly from the candidates $\hat{D}_{t-1}$ into replay buffer.

Reservoir [21] memory update algorithm conducts uniform sampling from $\hat{D}_{t-1}$. Specifically, the reservoir algorithm initializes the replay buffer indexed from 1 to L, containing the first L items $\{(x_1, y_1), (x_2, y_2), \ldots, (x_L, y_L)\}$ of the candidates. When updating replay buffer from the candidates, for each sample, the reservoir algorithm generates a random number m uniformly in $\{1, \ldots, len(\hat{D}_{t-1})\}$. If $m \in \{1, \ldots, L\}$, then the sample with the index m in the replay buffer is replaced with the sample $\hat{D}_{t-1}[m]$.



(a) Perturb per waveform **(b) Perturb per embedding**

Figure 1: *Block diagram of the native uncertainty approach and our proposed approach. Specifically, the naive approach adds perturbations to x by waveform and generates multiple waveform as $\hat{x}$. Our approach inputs the embedding e and generates perturbed embedding ê which means we only save the embedding. The output of the backbone of the model is calculated only once. "Compute $u(x)$" is to compute $u(x)$ by Eq. (3). The K refers to the number of the perturbations generated by perturb methods.*

Prototype [20] memory update algorithm selects the samples from $\hat{D}_{t-1}$ where the embedding of the classifier is close to the embedding mean of its own class. Specifically, the algorithm first groups the $\hat{D}_{t-1}$ into subsets as $D_c, c = 1 \ldots N^t$ by unique classes, where N^t denotes the total numbers of unique classes in the $\hat{D}_{t-1}$ set. Then the algorithm uses the current model to extract the embedding of the candidates for each D_c and calculates the class mean by the embedding as the average feature vector. For each class, the algorithm selects the samples of the candidates so that the average feature vector over the replay buffer provides best approximate to the average feature vector over all the samples of the corresponding class.

Uncertainty [22] memory update algorithm selects the sample by the uncertainty of the sample through the inference by the classification model. Specifically, the first step groups the $\hat{D}_{t-1}$ in the same way as the *prototype* algorithm introduced above. The second step estimates the uncertainty of each sample x in D_c. Predictive likelihood captures how well a model fits the data, with larger values indicating better model fit. Uncertainty score can be determined from predictive likelihood [31]. Following the derivation from [31], the predictive likelihood of a sample given by the model can be approximated by the Monte-Carlo (MC) integration [32] method with the model outputs of perturbed samples [24], which is defined as follows:

$$P(y = c \mid x) = \int p(y = c \mid \hat{x}) p(\hat{x} \mid x) d\hat{x}, \tag{3}$$

where x, $\hat{x}$, y denote an audio utterance of one class, the perturbed samples of x, and the label of x. Therefore, the uncertainty of the audio utterance x is formulated as $u(x)$:

$$u(x) \approx 1 - \frac{1}{K} \sum_{k=1}^{K} P(y = c \mid \hat{x}_k), \tag{4}$$

where K presents the number of the perturbations generated by perturb methods such as Audio Shift [33], Audio PitchShift [33] and Audio Colored Noise [34, 35]. A larger $u(x)$ indicates a smaller confidence of the model in predicting the perturbed samples. The third step selects L examples from D_c through descending the uncertainty $u(x)$ with the step size of $len(D_c) * C/L$, where L is the size of the replay buffer.

Previous research [24] demonstrated that the uncertainty memory update algorithm performs better than the other three algorithms on speech tasks such as keyword spotting. However, the computation cost of *Uncertainty* increases linearly with the number of perturbation operations.

2.2. Proposed MUA method (*Uncertainty++*)

As illustrated in Figure 1, the native uncertainty memory update algorithm requires to employ perturbation methods offline for the waveform of each sample to generate the perturbed samples first. In our proposed method, noisy perturbations are added to the pre-classifier embedding of the sample, and not to the waveform, so the output of the backbone of the model is calculated only once. Specifically, we propose a vector-wise perturbation method that adds noise with different intensities according to the variance of classifier's embedding. We denote the perturbed version of the classifier's embedding e as $\hat{e}$, which is computed as follows:

$$\hat{e} = e + U(-\frac{\lambda}{2}, \frac{\lambda}{2}) * std(e), \tag{5}$$

where $std(\cdot)$ stands for standard deviation, the function $U(a, b)$ represents the noise distributed uniformly from a to b, $U(a, b)$ is a vector with the same shape as e, and λ is a hyperparameter that controls the relative noise intensity.

By the vector-wise perturbation method, we generate the perturbed embedding $\hat{e}$ of the embedding e. Finally, we input $\hat{e}$ to the final classification layer of the model and output $P(y = c \mid \hat{e})$ which is used to compute the uncertainty as in Eq. (3). After the uncertainty is estimated, we select examples for replay as native approach. This method saves time by calculating the output of the backbone of the model only once. We also save the memory usage by replacing the perturbed raw data with the classifier's embedding which is of much smaller size as compared with the raw data.

3. EXPERIMENTS

3.1. Environmental sound classification model

For the on-device environmental sound classification model, we use BC-ResNet-Mod [29] which is an adaptation of the BC-ResNet [28] that achieves improved results on acoustic scene classification. The BC-ResNet paradigm works via repeatedly extracting spectral and then temporal features in series. Because these spectral features are of a lower dimension than the input, this model has fewer parameters than one that processes the waveform directly. Feature extraction is channel-wise, and both parameter reductions have negligible impact on performance [28]. For our experiments, we use BC-ResNet-Mod-4, which increases the input channel dimension to 80 before extracting spectral and temporal features.

3.2. Datasets

ESC-50 consists of 2000 five-second environmental audio recordings [27]. Data are balanced between 50 classes, with 40 examples per class, covering animal sounds, natural soundscapes, human sounds (non-speech), and ambient noises. The dataset has been prearranged into five folds for cross-validation.

DCASE 2019 Task 1 is an acoustic scene classification task, with a development set [26] consisting of 10-second audio segments from 10 acoustic scenes: airport, indoor shopping mall, metro station, pedestrian street, public square, the street with a medium level of traffic, traveling by tram, traveling by bus, traveling by an underground metro and urban park. In the development set, there are 9185 and 4185 audio clips for training and validation, respectively.

Table 1: *Accuracy (ACC) and Backward Transfer (BWT) in a comparative study of the proposed memory update algorithm.*

Method	DCASE 2019 Task 1		ESC-50	
	ACC ↑	BWT ↑	ACC ↑	BWT ↑
Finetune	0.205	-0.276	0.181	-0.307
Random	0.473	-0.115	0.225	-0.231
Reservoir	0.568	-0.096	0.430	-0.121
Prototype	0.559	-0.089	0.482	**-0.104**
Uncertainty	0.578	**-0.079**	0.477	-0.111
Uncertainty++	**0.581**	**-0.079**	**0.500**	-0.121

3.3. Experimental setup

Task setting To evaluate the performance of the proposed approach, we split the data into five tasks. Each task includes 2 new unique classes in DCASE 19 Task 1 and 10 new unique classes in ESC-50, which is unseen in previous tasks. To simulate the condition of edge devices, we set the buffer size L of examples as 500, 100 samples in DCASE 19 Task 1 and ESC-50 due to the memory limitation.

Implementation details The original audio clip is converted to 64-dimensional log Mel-spectrogram by using the short-time Fourier transform with a frame size of 1024 samples, a hop size of 320 samples, and a Hanning window. The classification network is optimized by the Adam [36] algorithm with the learning rate 1×10^{-3}. The batch size is set to 32 and the number of epochs is 50.

3.4. Evaluation metrics

We report performances in terms of the accuracy and forgetting metric. Specifically, the *Accuracy* (ACC) reports an accuracy averaged on learned classes after the entire training ends. The *Backward Transfer* (BWT) [37] evaluates accuracy changes on all previous tasks after learning a new task, indicating the forgetting degree. For measuring BWT, we first construct the matrix $R \in \mathbb{R}^{T \times T}$, where $R_{i,j}$ is the test classification accuracy of the model on task τ_j after observing the last sample from task τ_i. After the model finished learning about each task τ_i, we evaluate its BWT on all T tasks, which is formulated as:

$$BWT = \frac{1}{T-1} \sum_{i=1}^{T-1} R_{T,i} - R_{i,i}. \tag{6}$$

There exists negative BWT when learning about some task decreases the performance on some preceding task. A smaller value of BWT indicates a higher catastrophic forgetting.

3.5. Reference baselines

We built five baselines for comparisons. The *Finetune* training strategy adapts the BC-ResNet-Mod model for each new task without any continual learning strategies, as the lower-bound baseline. The four prior memory update algorithms of replay-based continual learning (i.e., *Random*, *Reservoir*, *Prototype*, *Uncertainty*) are introduced in Section 2.1. Specifically, at the perturbation stage of the uncertainty, we use two perturbation methods, namely, '*uncertainty-shift*', which

Table 2: *Accuracy (ACC) and Backward Transfer (BWT) in a comparative study of the proposed perturbation method. The K refers to the number of the perturbations generated by perturbation methods.*

Method	K	DCASE 2019 Task 1		ESC-50	
		ACC ↑	BWT ↑	ACC ↑	BWT ↑
Uncertainty-Shift	2	0.557	-0.101	0.461	**-0.111**
	4	0.575	-0.103	0.476	-0.118
	6	0.567	-0.079	0.477	-0.118
Uncertainty-Noise	2	0.560	-0.100	0.465	-0.118
	4	0.535	-0.104	0.473	-0.118
	6	0.578	-0.079	0.458	-0.120
Uncertainty++	2	0.571	-0.102	**0.500**	-0.121
	4	0.548	-0.103	0.481	-0.114
	6	**0.581**	**-0.079**	0.484	-0.119

includes Audio Shift and Audio PitchShift, and '*uncertainty-noise*' which refers to the Audio Colored Noise perturbation method.

4. RESULTS

4.1. Experiments on MUA methods

Table 1 presents the results on DCASE 2019 Task 1 and ESC-50 test set in terms of ACC and BWT. We compare the proposed *Uncertainty++* MUA method with five baselines. We observe that the *uncertainty* MUA method achieves better performance than the five baselines. Comparing with the best baseline *uncertainty*, we observe that the proposed *uncertainty++* method obtains 58.1% on classification accuracy which outperforms the existing MUA methods. In addition, we observe that the *Finetune* method achieves the worst ACC and BWT performance compared with other baselines, which indicates the issue of catastrophic forgetting.

We further analyze and summarize the performances of the proposed *uncertainty++* method compared with the *uncertainty* MUA method with different numbers of the perturbation methods in terms of ACC and BWT as shown in Table 2. The K refers to the number of the perturbations generated by perturb methods. Even with only two perturbation methods, our proposed method still outperforms other two baselines. We also observe that our method under two perturbations obtains the best performance on the ESC-50 test set. Such performance might be due to the small size of the ESC-50, therefore it is more sensitive to perturbations.

4.2. Comparative experiments on computation time for *Uncertainty* and *Uncertainty++*

We further report the Average Time for the proposed method when there is an increasing number of perturbations. The Average Time measures a relative time increase compared to training time in each task. As shown in Table 3, even with 6 perturbations, the Average Time of the *uncertainty++* is still less than 60s. This can be explained by the fact that our proposed method can limit the growth of the additional training time. We also observe that our proposed method outperforms other baselines in any number of perturbations, which indicates our proposed method is computationally more efficient. In addition, the average time of *uncertainty-shift* is much longer than others. Because the Audio Shift and Audio PitchShift perturbations takes more time than simply adding noise.

Table 3: *Average Time (s) in a comparative study of the proposed uncertainty++ method. The K refers to the number of the perturbations generated by perturbation methods.*

Method	K	Average Time (s) ↓
Uncertainty-Shift	2	1221.7
	4	2205.1
	6	2926.1
Uncertainty-Noise	2	246.2
	4	390.8
	6	506.3
Uncertainty++	2	44.0
	4	48.5
	6	55.1

5. CONCLUSIONS

In this work, we have presented *uncertainty++*, an efficient replay-based continual learning method for on-device environmental sound classification. Our method selects the historical data for the training by measuring the per-sample classification uncertainty on the embedding layer of the classifier. Experimental results on the DCASE 2019 Task 1 and ESC-50 datasets show that our proposed method outperforms the baseline continual learning methods on classification accuracy and computational efficiency. In future work, we plan to apply and adapt our approach to other on-device audio classification tasks such as audio tagging and sound event detection.

6. ACKNOWLEDGMENT

This work is partly supported by UK Engineering and Physical Sciences Research Council (EPSRC) Grant EP/T019751/1 "AI for Sound", a Newton Institutional Links Award from the British Council, titled "Automated Captioning of Image and Audio for Visually and Hearing Impaired" (Grant number 623805725), a Research Scholarship from the China Scholarship Council (CSC), and a PhD studentship from the Engineering and Physical Sciences Research Council (EPSRC) Doctoral Training Partnership EP/T518050/1. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising.

7. REFERENCES

[1] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *Proc. IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6.

[2] A. Singh, J. A King, X. Liu, W. Wang, and M. D. Plumbley, "Low-complexity CNNs for acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[3] A. Singh and M. D. Plumbley, "A passive similarity based cnn filter pruning for efficient acoustic scene classification," *arXiv preprint:2203.15751*, 2022.

[4] K. Choi, M. Kersner, J. Morton, and B. Chang, "Temporal knowledge distillation for on-device audio classification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 486–490.

[5] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola,

A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, “Low-complexity acoustic scene classification in DCASE 2022 Challenge,” *arXiv preprint:2206.03835*, 2022.

[6] R. Radhakrishnan, A. Divakaran, and A. Smaragdis, “Audio analysis for surveillance applications,” in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005.*, 2005, pp. 158–161.

[7] H. Liu, X. Liu, X. Mei, Q. Kong, W. Wang, and M. D. Plumbley, “Surrey system for DCASE 2022 task 5 : Few-shot bioacoustic event detection with segment-level metric learning technical report,” DCASE2022 Challenge, Tech. Rep., June 2022.

[8] S. Kiranyaz, A. F. Qureshi, and M. Gabbouj, “A generic audio classification and segmentation approach for multimedia indexing and retrieval,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 3, pp. 1062–1081, 2006.

[9] J. Salamon and J. P. Bello, “Deep convolutional neural networks and data augmentation for environmental sound classification,” *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[10] J. Sun, X. Liu, X. Mei, J. Zhao, M. D. Plumbley, V. Kılıç, and W. Wang, “Deep neural decision forest for acoustic scene classification,” *arXiv preprint:2203.03436*, 2022.

[11] N. Hou, C. Xu, E. S. Chng, and H. Li, “Domain adversarial training for speech enhancement,” in *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2019, pp. 667–672.

[12] N. Hou, C. Xu, J. T. Zhou, E. S. Chng, and H. Li, “Multi-task learning for end-to-end noise-robust bandwidth extension,” in *Proc. Interspeech*, 2020, pp. 4069–4073.

[13] M. McCloskey and N. J. Cohen, “Catastrophic interference in connectionist networks: The sequential learning problem,” in *Psychology of Learning and Motivation*, 1989, vol. 24, pp. 109–165.

[14] A. Awasthi and S. Sarawagi, “Continual learning with neural networks: A review,” in *Proc. the ACM India Joint International Conference on Data Science and Management of Data*, 2019, pp. 362–365.

[15] H. Zhang, M. Shen, Y. Huang, Y. Wen, Y. Luo, G. Gao, and K. Guan, “A serverless cloud-fog platform for DNN-based video analytics with incremental learning,” *arXiv preprint:2102.03012*, 2021.

[16] Y. Huang, H. Zhang, Y. Wen, P. Sun, and N. B. D. Ta, “Modelci-e: Enabling continual learning in deep learning serving systems,” *arXiv preprint:2106.03122*, 2021.

[17] 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,” *the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[18] F. Zenke, B. Poole, and S. Ganguli, “Continual learning through synaptic intelligence,” in *Proc. International Conference on Machine Learning*, 2017, pp. 3987–3995.

[19] Y.-C. Hsu, Y.-C. Liu, A. Ramasamy, and Z. Kira, “Re-evaluating continual learning scenarios: A categorization and case for strong baselines,” *arXiv preprint:1810.12488*, 2018.

[20] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, “iCaRL: Incremental classifier and representation learning,” in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[21] J. S. Vitter, “Random sampling with a reservoir,” *ACM Transactions on Mathematical Software (TOMS)*, vol. 11, no. 1, pp. 37–57, 1985.

[22] J. Bang, H. Kim, Y. Yoo, J.-W. Ha, and J. Choi, “Rainbow memory: Continual learning with a memory of diverse samples,” in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8218–8227.

[23] Y. Huang, N. Hou, and N. F. Chen, “Progressive continual learning for spoken keyword spotting,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 7552–7556.

[24] Y. Xiao, N. Hou, and E. S. Chng, “Rainbow keywords: Efficient incremental learning for online spoken keyword spotting,” *arXiv preprint:2203.16361*, 2022.

[25] Z. Wang, C. Subakan, E. Tzinis, P. Smaragdis, and L. Charlin, “Continual learning of new sound classes using generative replay,” in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 308–312.

[26] T. Heittola, A. Mesaros, and T. Virtanen, “TAU Urban Acoustic Scenes 2019, Development dataset,” Mar. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2589280

[27] K. J. Piczak, “Esc: Dataset for environmental sound classification,” in *Proc. the ACM international conference on Multimedia*, 2015, pp. 1015–1018.

[28] B. Kim, S. Chang, J. Lee, and D. Sung, “Broadcasted residual learning for efficient keyword spotting,” *arXiv preprint:2106.04140*, 2021.

[29] B. Kim, S. Yang, J. Kim, and S. Chang, “QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design,” DCASE2021 Challenge, Tech. Rep., June 2021.

[30] Y.-C. Hsu, Y.-C. Liu, A. Ramasamy, and Z. Kira, “Re-evaluating continual learning scenarios: A categorization and case for strong baselines,” *arXiv preprint:1810.12488*, 2018.

[31] Y. Gal and Z. Ghahramani, “Dropout as a Bayesian approximation: Representing model uncertainty in deep learning,” 2016.

[32] T. Kloek and H. K. Van Dijk, “Bayesian estimates of equation system parameters: An application of integration by monte carlo,” *Econometrica: Journal of the Econometric Society*, pp. 1–19, 1978.

[33] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, “Audio augmentation for speech recognition,” in *Proc. Interspeech*, 2015.

[34] I. Jordal, S. ES, H. BREDIN, K. Nishi, F. Lata, H. C. Blum, P. Manuel, akash raj, K. Choi, FrenchKrab, P. Żelasko, amiasato, M. L. Quatra, and E. Schmidbauer, “asteroid-team/torch-audiomentations: v0.11.0,” June 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6778064

[35] S. Kamath and P. Loizou, “A multi-band spectral subtraction method for enhancing speech corrupted by colored noise,” in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2002.

[36] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint:1412.6980*, 2014.

[37] D. Lopez-Paz and M. Ranzato, “Gradient episodic memory for continual learning,” in *Proc. Advances in neural information processing systems*, vol. 30, 2017.