IMPROVING FEATURE GENERALIZABILITY WITH MULTITASK LEARNING IN CLASS INCREMENTAL LEARNING

Dong Ma[†,1,2], *Chi Ian Tang*[†,1], *Cecilia Mascolo*[1]

[1]University of Cambridge, [2]Singapore Management University
{dm878, cit27, cm542}@cam.ac.uk

ABSTRACT

Many deep learning applications, like keyword spotting [1, 2], require the incorporation of new concepts (classes) over time, referred to as Class Incremental Learning (CIL). The major challenge in CIL is catastrophic forgetting, i.e., preserving as much of the old knowledge as possible while learning new tasks. Various techniques, such as regularization, knowledge distillation, and the use of exemplars, have been proposed to resolve this issue. However, prior works primarily focus on the incremental learning step, while ignoring the optimization during the base model training. We hypothesise that a more transferable and generalizable feature representation from the base model would be beneficial to incremental learning.

In this work, we adopt multitask learning during base model training to improve the feature generalizability. Specifically, instead of training a single model with all the base classes, we decompose the base classes into multiple subsets and regard each of them as a task. These tasks are trained concurrently and a shared feature extractor is obtained for incremental learning. We evaluate our approach on two datasets under various configurations. The results show that our approach enhances the average incremental learning accuracy by up to 5.5%, which enables more reliable and accurate keyword spotting over time. Moreover, the proposed approach can be combined with many existing techniques and provides additional performance gain.

Index Terms— Class Incremental Learning, Continual Learning, Multitask Learning, Keyword Spotting

1. INTRODUCTION

Recently, deep learning has enabled the boom of a variety of applications such as face recognition [3] and keyword spotting [1, 2]. Albeit remarkable performance, deep learning models are usually built upon a fixed dataset, lacking the ability and flexibility of adapting to sequentially incoming data (of the same class or new classes) [4]. For example, in keyword spotting, the initial model is built based on a pre-defined keyword set. When the user intends to add new keywords over time, the data-hungry nature of deep learning incurs two challenges for model update. First, storing the previous data might require substantial memory. Second, retrieving old data and training a new model from scratch requires considerable time and computational resources. A research area, class incremental learning (CIL), which tries to retain the acquired knowledge while learning new concepts, has been initiated to address these challenges [5].

Typically, CIL consists of two stages, as shown in Figure 1. The base model training stage utilizes the full dataset at hand to train a base model. During the incremental learning stage, the base model will be fine-tuned with new data received over time (and a part of the old data). A critical problem in CIL is *catastrophic forgetting* [4, 5, 6, 7, 8, 9], that is, the model tends to be overfitted on the new class data, while forgetting previously acquired knowledge, thereby degrading the inference performance on old classes. This originates from the imbalance between the old and new class data as none or only part of the old data is retained to save storage during fine-tuning [10, 11]. Moreover, the data imbalance issue also implies bias at the classification layer: old samples are more likely to be classified as new classes [12].

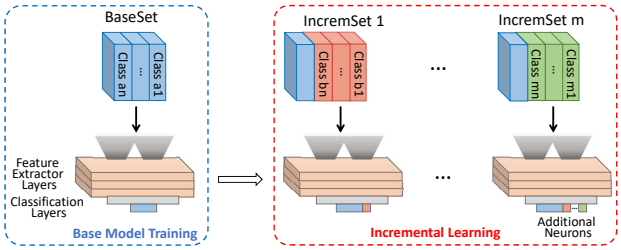


Fig. 1. Illustration of typical class incremental learning.

To deal with catastrophic forgetting, researchers have proposed different techniques, which can be divided into three categories. The initial attempt to resolve catastrophic forgetting is regularization. Specifically, this method first locates the important model parameters that contribute more to the final classification by using certain metrics such as the fisher information [7] and output gradients [6]. Then, an additional term is applied to the loss function to restrict the changes of these important weights. Later, knowledge distillation [13], which transfers knowledge between different domains or networks, has been adopted to retain the old knowledge when learning a new task [9, 14]. The output logits of new samples on the old model will be recorded first, and an extra loss term is added to minimize its difference with the output logits from the new model during incremental training. These two approaches have been proven to have poor performance in CIL [15]. Most recently, exemplar-based approaches were proposed, where a small portion of old samples are selected (randomly or with herding technique [16, 17, 18]) and stored. In the incremental learning step, the exemplar set is combined with new data for model updates, which shows great performance in retaining old knowledge and learning new concepts. The state-of-the-art incremental learning performance is achieved with a combination of different techniques [19].

However, we identified that *all the existing approaches focus on the incremental learning step, while ignoring the optimizations in the base model training stage*. Intuitively, to retain the old knowledge while learning new concepts, we would that expect the feature representation (or embedding) produced by the base model to be general to the new classes as well [19]. In other words, if we can train a

† Authors with equal contribution in alphabetical order.

more representative and transferable base model, we can alleviate the catastrophic forgetting issue during incremental learning. This assumption is tenable due to the fact that for a given neural network and accuracy, there exist multiple sets of weights, i.e., training a neural network multiple times with the same setting results in different sets of model weights. So, the question is *how to obtain a set of weights that is more transferable to new classes*.

In this work, *we propose the use of multitask learning during the base model training to improve the generalizability of embeddings*. Multitask learning involves learning multiple tasks in parallel while using a shared representation [20]. If we define different subsets of base classes as different classification tasks, the model would see different combinations of the base classes. As a result, the model is forced to learn a set of weights based on different views of the dataset, which would be more transferable to new classes. Moreover, the number of training epochs needed for incremental learning is decided empirically in current approaches. We introduce a validation set and the early stopping technique to avoid underfitting or overfitting in practical scenarios. Experimenting with the Google Speech Commands (GSC) and UrbanSound8K datasets, we demonstrated that our approach can further enhance the average classification accuracy by up to 5.5%, which enables more reliable and accurate keyword spotting over time. Furthermore, our approach is compatible with many other existing approaches and adds extra performance gains over them as it works on the base model training stage.

2. PRIMER

2.1. Related Work

Existing literature on CIL mainly addresses two research problems, i.e., catastrophic forgetting of old knowledge and data imbalance between old and new classes.

Catastrophic forgetting: refers to the fact that the model forgets previously attained knowledge on old classes when learning new concepts with new class data. Previous works to combat catastrophic forgetting can be categorized into two groups - without and with samples of old classes. Techniques such as parameter control [6, 7, 8], knowledge distillation [13], exemplar replay [16, 17, 18, 21, 22, 23], and generative adversarial network [24] are proposed in the literature, as discussed in the Introduction.

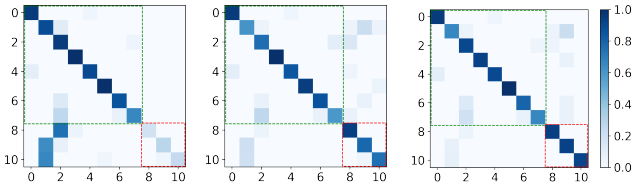
Data imbalance: no matter with or without old samples, there exists a huge imbalance between old and new samples during incremental learning stage. As a result, the model will be overfitted to the new classes. To address it, BiC [10] adds an extra bias correction layer to correct the model's outputs. WA-MDF [12] aligns the norm of new class weight vectors to that of the old class weight vectors and [11] applied cosine normalization in the classification layer.

Overall, the state-of-the-art approaches combine multiple techniques such as knowledge distillation and data balancing to achieve optimal performance [19].

2.2. Motivation

For each line of work, we identified certain limitations that motivate this work.

First, existing approaches that deal with catastrophic forgetting all focus on the incremental learning stage, while ignoring the importance of the base model training. In particular, we observed that training a neural network for multiple times under the same setting[1] will result in different sets of weights (or embeddings), i.e., the model weights are not unique. This observation raises a question, *among different sets of model weights, which one is more beneficial for incremental learning?* Since less transferable embeddings will force the model to alter its weights significantly to learn new concepts, the previously attained knowledge will be destroyed, leading to more severe catastrophic forgetting. Thus, we hypothesise that *it is possible to alleviate catastrophic forgetting by training a more transferable base model.*



(a) Epoch 15 (b) Epoch 45 (c) Epoch 100

Fig. 2. Comparison of confusion matrix after training for (a) 15 (acc old = 0.91, acc new = 0.24), (b) 45 (acc old = 0.83, acc new = 0.82), and (c) 100 (acc old = 0.86, acc new = 0.92) epochs. The green and red box represent old and new classes, respectively.

Second, during incremental learning, existing works usually report an empirical value for training epochs[2] and this number varies a lot for the same dataset. For instance, on the CIFAR-100 datset, the number of training epoch for BiC [10], WA-ADB [12] and iCarl [23] is set to 250, 180, and 70 respectively. With the increase of training epochs, the model tends to shift to new classes due to data imbalance between old classes and new classes. Figure 2 compares the confusion matrix of a CIL task at different training epochs, where class 0-7 are old classes and class 8-10 are new classes. We can observe that when the number of training epoch is small (i.e., 15), the model has not acquired enough knowledge about the new classes and the accuracy on new classes is poor. With more training epochs, the performance on new classes increases gradually, and the model becomes overfitted with excessive epochs (i.e., 100). Consequently, setting a proper value for training epochs is critical to deal with data imbalance in CIL.

3. METHODOLOGY

Motivated by the discussion above, we introduce *multitask learning* to CIL to improve the generalizability of feature representations. As shown in Figure 3, the full base dataset is decomposed into multiple subsets and they form a multitask setting, where the classification of each set is regarded as a task. These tasks are trained concurrently with a shared representation. After multitask training, the model backbone and the largest head (corresponding to full base dataset) is forwarded to the incremental learning step, where the state-of-the-art techniques can be directly applied.

3.1. Intuition of Multitask Training

Multitask learning forces the model to simultaneously solve multiple tasks at once [20], and has been used to help improve the generalization of the model in solving all tasks, by preventing the model from overfitting to any particular task [25]. In the context of CIL, generalizability of the model is crucial, since it is necessary for the model

[1] Without changing any hyper-parameter and simply run the same code again to achieve a similar accuracy.

[2] Such empirical value is obtained with the full dataset of the new classes, which is unavailable in practical CIL setting.

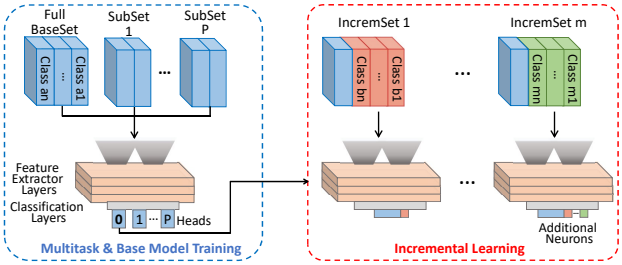


Fig. 3. Illustration of multitask learning based CIL.

to retain previous knowledge while learning about new classes. We designed a multitask learning scheme that aims to simulate the incremental learning steps at the base model training stage, by dividing the dataset into different subsets. For example, with base classes such as 'air conditioner', 'car horn', 'children playing' and 'drilling', we can create different classification tasks by taking clusters of them, including {'air conditioner', 'car horn'}, {'air conditioner', 'drilling'} and {'air conditioner', 'car horn', 'children playing', 'drilling'}. This grouping of the base learning classes aims to make the model find a solution which can solve all these classification tasks at once, instead of overfitting to the whole base dataset. As a result, the generalizability of the base model is improved, leading to better performance in the incremental learning stage. This particular setup is analogous to the incremental learning stage, where the model is required to perform well on different sets of classes.

3.2. Multitask Creation

Important design choices that arise from adopting multitask training are: (1) the number of tasks, and (2) the cluster of classes that each task corresponds to. If we have N target classes in the dataset used in base model training, there are $2^N - 1$ distinct valid tasks (which equals to the number of subsets excluding the empty set). With the number of task being 1 and the cluster being the entire set of classes, we have the conventional setup of a fully-supervised training on the base dataset. The number of distinct tasks grows very rapidly as N increases, so it is infeasible to train on all of them, and a design choice should be made. With the number of classes being closer to N, the task itself is harder. However, restricting all the tasks to have high number of classes reduces the diversity of tasks presented to the model, and it may not help in improving the generalizability of the solution that the model converges to. In this work, we explore different choices of tasks, along two directions: (1) tasks with different number of classes (such as 5, 4, and 3 classes for each task), and (2) those with different subsets of classes but the same number of classes (such as 5, 4, 4 and 4 classes, but each task is a distinct subset).

3.3. Learning Rate Scheduler

One of the most important hyperparameters affecting the amount of knowledge retention in CIL is the learning rate during incremental steps, in which the models are fine-tuned. Some existing works adopts a cosine annealing for training with a large starting learning rate (0.01) [19]. We observed that this large starting learning rate significantly changes the weights of the neural network, and this could make knowledge retention difficult. On the other hand, adopting a fixed small learning rate causes training to be slow.

As a result, we propose a two-step fine-tuning strategy to mitigate this. First, the majority of the up-stream layers of the neural network are frozen, while the down-stream layers which include the classification layer with the newly added neurons are trained with a relatively large learning rate. After the down-stream layers, and the new neurons in particular, converge to a reasonable solution to both new and old classes, the up-stream layers are unfrozen and the entire network is fine-tuned with a small learning rate. An early stopping mechanism is also adopted in our scheme because it is difficult to find the balance between overfitting and underfitting when using a fixed number of training epochs (which is commonly adopted in previous works).

4. EVALUATION

4.1. Experiment Setting

Dataset: We evaluated the proposed approach on two audio datasets. UrbanSound8K dataset [26] consists of 10 different environment sound events such as drilling, car horn, street music, etc. There are 8,732 audio clips sampled at 22 kHz, where each clip lasts for 3-4 seconds. Following [27], we extracted four audio features (Log-mel spectrogram, chroma, tonnets, and spectral contrast) using the first 3-seconds of a clip. As a result, the created input has a size of 128×85, where 128 represents the number of frames and 85 represents aggregated feature size of the four audio features. Google Speech Commands (GSC) [26] is a widely used dataset in keyword spotting. We pick the 20 core keywords as the classes and each class contains 3,200 1-second clip. Log-mel spectrogram features have been extracted and a 32×32 input is created.

For UrbanSound8K dataset, we set the number of base class and incremental class as 4 and 2 respectively, i.e., 4-2-2-2. Similarly, GSC dataset is split to 5-3-3-3-3-3. For both datasets, the (train, test, validation) splitting ratio is set to (0.7, 0.2, 0.1). We allocate a fixed memory to store K (default 100 for UrbanSound8K and 200 for GSC) exemplars for all previously seen classes.

Model Architecture: We designed a convolutional neural network to classifier the audio events. The network consists of four Conv2D layers with ReLU activation, each followed by a BatchNormalization layer. Then, a Dropout (0.5) layer and an AveragePooling layer is connected before the final classification layer. Stochastic gradient descent is used as the optimizer. All the code is implemented in PyTorch and run on a NVIDIA GeForce RTX 2080 GPU. We use the hard parameter sharing approach for our multitask learning, where the hidden layers for different tasks are shared and remain exactly the same and only the classification layer is trained separately.

Baseline: We compare our work to the method proposed by Mittal et al. [19] which reported state-of-the-art results for CIL. Specifically, [19] first utilizes the cross entropy (CE) loss and knowledge distillation (KD) loss on new classes to learn new knowledge. Then, it constructs a small but balanced exemplar set (including current incremental classes) to correct the bias and preserve old knowledge (with CE loss and KD loss). For more details, please refer to the original paper [19]. We adopted the same incremental learning strategy as in [19] and the difference only comes from the base model training, i.e., single task ([19]) vs. multitask (ours).

4.2. Results

Impact of Task Creation: First, we systematically explored the configurations of different tasks along two directions: (1) tasks with different number of classes, and (2) those with different subsets of classes. Figure 4 demonstrates the change in model accuracy as we vary the configuration of tasks, on the GSC dataset. From Figure 4(a), we could infer that adding more tasks with the same number of

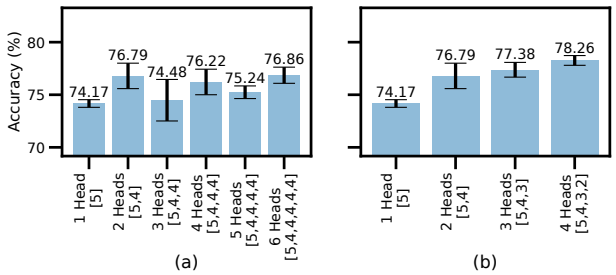


Fig. 4. Impact of task creation.

Table 1. Time overhead when training with different number and choice of tasks/heads.

Heads	Training Time per Epoch (s)	Heads	Training Time per Epoch (s)
[5]	2.16	[5,4,4]	4.79
[5,4]	3.41	[5,4,4,4]	5.35
[5,4,3]	4.65	[5,4,4,4,4]	6.44
[5,4,3,2]	5.44	[5,4,4,4,4,4]	7.42

classes but different subsets did not lead to consistent performance improvements. On the other hand, from Figure 4(b), we could see that increasing the number of tasks with different classes led to a consistent increase of performance, up to 5%, although a diminishing effect could be seen when we add more tasks.

We hypothesise that having many similar tasks may not offer much help in improving the model's generalizability, while having more distinct tasks is better at doing that. This is because our multitask training objective is to allow the model to see wider and more diverse combinations of the base classes, so that it can be effectively extended to new classes during incremental learning. Consequently, if the tasks are too similar (i.e., huge overlapping of classes), the model might not be able to learn more generalizability. For example, in the extreme case, if all the tasks contain the same classes, our approach degrades to single task learning.

For a given set of base classes, a good construction of multiple tasks is critical for our approach. In this work, we focused on the number of classes and tasks in multitask learning, without utilizing the prior knowledge about the dataset. For example, grouping classes based on their semantic meaning might be helpful in creating high-quality tasks. We will explore it in the future.

In terms of time overhead, from the results shown in Table 1, we could observe consistent increase in training time per epoch when the number of tasks was increased, and when the number of classes in each task was increased.

Overall, the results demonstrated that having more than a single task helps in improving the model's performance, especially when the tasks are more distinct from each other. We will therefore select '4 heads [5, 4, 3, 2]' as the multitask setting for the rest of the evaluation.

Impact of Exemplar Quantity: As the number of exemplars available at the incremental learning stage can have a significant impact on the performance, we further conducted a set of experiments with varying quantity of exemplars: 20, 50, 100, 200, 300 and 400 for Google Speech Commands (GSC) and 10, 20, 50, 100 and 200 for UrbanSound8K (US8K). Figure 5 shows the accuracy of the model across different amounts of exemplars. We could see a general trend of improving performance as the number of exemplars was increased, with a diminishing effect. The performance of models starts to plateau with around 200 exemplars on the GSC dataset, and around 100 exemplars on US8K.

Comparison with Baseline: To compare with the state-of-the-art

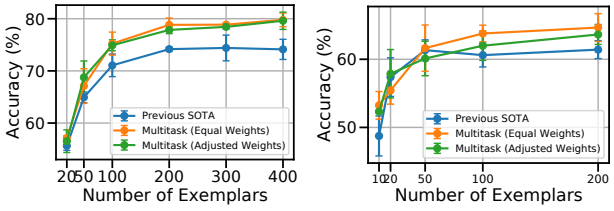


Fig. 5. Impact of exemplar quantity.

Table 2. Effect of different losses on the incremental learning performance. CE refers to Cross Entropy, KD refers to Knowledge Distillation, N refers to New samples, and O refers to Old samples (exemplars).

# of class	5	8	11	14	17	20	Avg
CE_N	96.97	60.23	43.79	35.73	29.46	26.22	**39.09**
CE_N+KD_N	97.08	60.75	43.34	37.43	36.20	34.85	**42.52**
CE_N+CE_O	97.25	88.77	79.07	72.88	72.82	57.27	**74.16**
CE_N+CE_O +KD_N	96.78	84.65	78.27	77.91	73.60	72.55	**77.39**
CE_N+CE_O +KD_O	97.12	85.72	80.64	78.99	74.30	71.93	**78.32**
CE_N+CE_O+ KD_N+KD_O	97.35	87.92	81.47	77.66	73.80	73.27	**78.82**

method, we conducted the same set of experiments mentioned in the previous section, using only a single task as the baseline. From Figure 5, we could see that our method started with similar performance as the previous state-of-the-art method when very few exemplars were allowed, and showed a consistent improvement of up to 5.5% in accuracy compared to the baseline as the amount of exemplars increased on GSC, and up to 3.2% on US8K.

Impact of Losses: To understand which technique contributes most to the incremental learning performance, we created different variants of the approach by selecting different losses. Table 2 presents the results at different incremental steps and the average accuracy for incremental learning. We can observe that without exemplars (first two rows), the accuracy drops substantially (by 40%) compared to the optimal performance. Knowledge distillation merely improves the accuracy by 3% (row 2-row 1, or row 4-row 3). Overall, the performance gain is dominated by the use of exemplars and knowledge distillation provides limited accuracy enhancement.

5. FINAL REMARKS

In this work, we first identified that a more transferable feature representation of the base model might be beneficial for incremental learning. Then, we introduced multitask learning to the base model training stage to improve the generalizability of representations. With two audio datasets, we explored the impact of multitask creation, exemplar quantity, and different losses. The results show that our approach improves the average incremental accuracy by up to 5.5%. A thorough comparison with more baselines is planed as a future work. Our work opens the door to improving the quality of base model in incremental learning, which motivates the exploration of various generalization techniques in the future.

6. ACKNOWLEDGEMENT

This work is partially supported by Nokia Bell Labs through their donation for the Centre of Mobile, Wearable Systems and Augmented Intelligence to the University of Cambridge. This work is also supported by ERC through Project 833296 (EAR).

7. REFERENCES

[1] Guoguo Chen, Carolina Parada, and Georg Heigold, "Small-footprint keyword spotting using deep neural networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4087–4091.

[2] Igor Szöke, Petr Schwarz, Pavel Matejka, Lukás Burget, Martin Karafiát, Michal Fapso, and Jan Cernockỳ, "Comparison of keyword spotting approaches for informal continuous speech.," in *Interspeech*, 2005, pp. 633–636.

[3] Weihong Wang, Jie Yang, Jianwei Xiao, Sheng Li, and Dixin Zhou, "Face recognition based on deep learning," in *International Conference on Human Centered Computing*. Springer, 2014, pp. 812–820.

[4] Tom Diethe, Tom Borchert, Eno Thereska, Borja Balle, and Neil Lawrence, "Continual learning in practice," *arXiv preprint arXiv:1903.05202*, 2019.

[5] Gido M Van de Ven and Andreas S Tolias, "Three scenarios for continual learning," *arXiv preprint arXiv:1904.07734*, 2019.

[6] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 139–154.

[7] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka 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.

[8] Friedemann Zenke, Ben Poole, and Surya Ganguli, "Continual learning through synaptic intelligence," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3987–3995.

[9] Zhizhong Li and Derek Hoiem, "Learning without forgetting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[10] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu, "Large scale incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 374–382.

[11] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin, "Learning a unified classifier incrementally via rebalancing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 831–839.

[12] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia, "Maintaining discrimination and fairness in class incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13208–13217.

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[14] Peng Zhou, Long Mai, Jianming Zhang, Ning Xu, Zuxuan Wu, and Larry S Davis, "M2kd: Multi-model and multi-level knowledge distillation for incremental learning," *arXiv preprint arXiv:1904.01769*, 2019.

[15] Saurav Jha, Martin Schiemer, Franco Zambonelli, and Juan Ye, "Continual learning in sensor-based human activity recognition: an empirical benchmark analysis," *arXiv preprint arXiv:2104.09396*, 2021.

[16] David Lopez-Paz and Marc'Aurelio Ranzato, "Gradient episodic memory for continual learning," *Advances in neural information processing systems*, vol. 30, pp. 6467–6476, 2017.

[17] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan, "Remind your neural network to prevent catastrophic forgetting," in *European Conference on Computer Vision*. Springer, 2020, pp. 466–483.

[18] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid, "Memory-efficient incremental learning through feature adaptation," in *European Conference on Computer Vision*. Springer, 2020, pp. 699–715.

[19] Sudhanshu Mittal, Silvio Galesso, and Thomas Brox, "Essentials for class incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3513–3522.

[20] Rich Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[21] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun, "Mnemonics training: Multi-class incremental learning without forgetting," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, 2020, pp. 12245–12254.

[22] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari, "End-to-end incremental learning," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 233–248.

[23] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[24] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[25] Yu Zhang and Qiang Yang, "A survey on multi-task learning," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2021.

[26] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello, "A dataset and taxonomy for urban sound research," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 1041–1044.

[27] Young D Kwon, Jagmohan Chauhan, and Cecilia Mascolo, "Fasticarl: Fast incremental classifier and representation learning with efficient budget allocation in audio sensing applications," *arXiv preprint arXiv:2106.07268*, 2021.