PROGRESSIVE CONTINUAL LEARNING FOR SPOKEN KEYWORD SPOTTING

Yizheng Huang[⋆] *Nana Hou*[†] *Nancy F. Chen*[⋆]

[⋆]Institute for Infocomm Research, A*STAR, Singapore
[†]Nanyang Technological University, Singapore

ABSTRACT

Catastrophic forgetting is a thorny challenge when updating keyword spotting (KWS) models after deployment. To tackle such challenges, we propose a progressive continual learning strategy for small-footprint spoken keyword spotting (PCL-KWS). Specifically, the proposed PCL-KWS framework introduces a network instantiator to generate the task-specific sub-networks for remembering previously learned keywords. As a result, the PCL-KWS approach incrementally learns new keywords without forgetting prior knowledge. Besides, the proposed keyword-aware network scaling mechanism of PCL-KWS constrains the growth of model parameters while achieving high performance. Experimental results show that after learning five new tasks sequentially, our proposed PCL-KWS approach archives the new state-of-the-art performance of 92.8% average accuracy for all the tasks on *Google Speech Command* dataset compared with other baselines.

Index Terms— Continual learning, Incremental learning, Keyword spotting

1. INTRODUCTION

Spoken keyword spotting (KWS) aims to identify specific keywords in the audio input. It serves as a primary module in many real-word applications, such as Apple Siri and Google Home. Current approaches to keyword spotting are fueled by deep learning models [1, 2, 3, 4], which are usually trained with limited keywords in compact model for lower computation and smaller footprint [3, 5]. Therefore, the performance of the KWS model trained by the source-domain data may degrade significantly when confronted with unseen keywords of the target-domain at run-time.

To alleviate such problem, prior work [6, 7, 8] utilizes few-shot fine-tuning [9] to adapt KWS models with training data from the target-domain for new scenarios. The limitations of such an approach are two-fold. First, performance on data from the source domain after adaptation could be poor, which is also known as *catastrophic forgetting* problem [10]. Second, the fine-tuning process usually requires a large pre-trained speech embedding model to guarantee the performance, requiring higher memory cost and is undesirable for small-footprint KWS scenarios.

Continual learning (CL) [11, 12, 13] addresses the catastrophic forgetting issue when adapting pre-trained model with target-domain data. Some efforts in continual learning have been applied to speech recognition task [14, 15]. We can categorize existing CL methods based on replay buffer (the requirement of the extra memory to store data/parameters) into two categories: regularization-based methods and replay-based methods. Regularization-based methods protects the parameters learned from previous tasks by loss regularization (e.g., EWC [16], SI [17]). Replay-based methods require a buffer to store 1) historical data as the replay exemplars (e.g., Rehearsal [18, 19], GEM [20]), or 2) model weights to isolate the parameters learned by different tasks [21, 22, 23].

In this paper, we first investigate and analyze the prior four continual learning strategies (EWC, SI, NR, GEM) for the keyword spotting task. Then, we propose a progressive continual learning strategy for small-footprint keyword spotting (PCL-KWS) to alleviate the catastrophic forgetting problem when adapting the model trained by source-domain data with the target-domain data. Specifically, the proposed PCL-KWS includes several task-specific sub-networks to memorize the knowledge of the previous keywords. Then, a keyword-aware network scaling mechanism is introduced to reduce the network parameters. As a result, the proposed PCL-KWS could preserve the high accuracy of the old tasks when learning the new tasks without increasing model parameters significantly. The experimental results show that PCL-KWS achieves state-of-the-art performance with only a few extra parameters and no buffers.

2. PCL-KWS ARCHITECTURE

This section first describes four continual learning strategies, including regularization-based methods and reply-based methods. The proposed progressive continual learning for keyword spotting (PCL-KWS) are then introduced.

2.1. Continual Learning Workflow

The workflow of conducting continual learning for keyword spotting is to receive T landed keyword learning tasks sequentially and optimize performances on all the T tasks without catastrophic forgetting. For each task τ_t with $t \leq T$, we have

a training dataset including pairs (x_t, y_t), where x_t are the audio utterances following the distribution D^T and y_t are the keyword labels. We aim to minimize the total loss function $\mathcal{L}_{tot}$, which sums the sub-losses $\mathcal{L}_{kws}$ of all T tasks formulated as:

$$\mathcal{L}_{tot} = \sum_{t=0}^{T} \mathbb{E}_{(x_t,y_t)\sim D^T} \left[\mathcal{L}_{kws}(F_t(x_t;\theta^t), y_t) \right] \quad (1)$$

where $\mathcal{L}_{kws}$ denotes the cross-entropy loss and $F_t(x_t;\theta^t)$ is the KWS model with parameters θ^t. During training, we seek the optimal parameters $\theta^{t'}$ for all the T tasks. It is a challenging task since the parameters learned from the previous task τ_t are easily overwritten after learning the new task τ_{t+1}, incurring catastrophic forgetting.

To robustify performance, we introduce regularization-based methods and replay-based methods for the KWS task.

2.1.1. *Regularization-based Methods*

The regularization-based methods protect important parameters trained on previous tasks from being overwritten by adding regular terms to the loss function $\mathcal{L}_{kws}$. We introduce two commonly-used regularization terms: Elastic Weight Consolidation (EWC) [16] and Synaptic Intelligence (SI) [17], which are explored as baselines in this work.

EWC first calculates the parameter importance Ω_i^t for i^{th} parameter of θ^t in each task τ_t. Ω_i^t acts as the importance regularization to constrain more important parameters as close to those of previous $t-1$ tasks, while updating parameters of less importance with the t^{th} new task. The optimized loss function is formulated as:

$$\mathcal{L}'_{kws} = \mathcal{L}_{kws}(F_t(x_t;\theta^t), y_t) + \frac{\lambda}{2} \sum_i \Omega_i^t (\theta_i^t - \theta_i^{t-1})^2, \quad (2)$$

where the importance coefficient Ω_i^t is determined by the fisher information matrix (FIM):

$$\Omega_i^t = \mathbb{E}_{(x_t,y_t)\sim D^T} \left[\left(\frac{\delta \mathcal{L}_{kws}}{\delta \theta_i^t} \right)^2 \right]. \quad (3)$$

SI is derived from EWC, which assigns the parameter importance Ω_i^t in a different manner. It utilizes the gradient ω_i^t, calculated from the KWS loss $\mathcal{L}_{kws}$, to estimate the i^{th} parameter importance at each training step,

$$\Omega_i^{t+1} = \Omega_i^t + \frac{\omega_i^t}{\left(\theta_i^t - \theta_i^{t-1}\right)^2 + \varepsilon} \quad (4)$$

where the damping parameter ε is used to avoid division by zero, making the training process more stable.

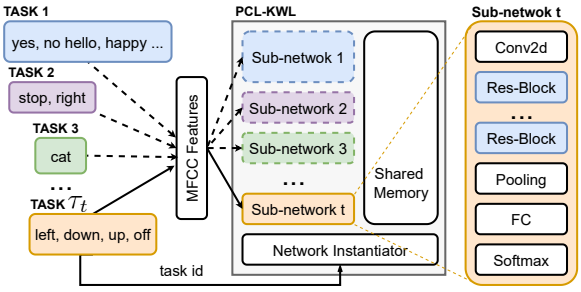

Fig. 1: *Block diagram of the proposed PCL-KWS.*

2.1.2. *Replay-based Methods*

The replay-based methods utilize an in-memory buffer to store the historical training samples to maintain the accuracy of the previous $t-1$ tasks. We adopt Naive Rehearsal (NR) [18] and Gradient Episodic Memory (GEM) [20] to demonstrate replay-based methods.

NR stores the randomly selected training samples from previous $t-1$ learned task $\{\tau_0, \tau_1 ... \tau_{t-1}\}$ into replay buffer, and builds the training data D'_t of the task τ_t formulated as:

$$D'_t = \xi(D_1, D_2, ..., D_{t-1}) \cup D_t, \quad 0 < \xi \leq 1 \quad (5)$$

where the $\xi\%$ denotes the percentage of utilized historical data, and D_t is the incoming dataset for the new task.

GEM calculates the gradients g_k on the buffered data of task τ_k with $k \leq t$. If the current gradient g_t of the task τ_t degrades the performances on any previous tasks, it projects g_t to the gradient $\tilde{g}$, which has the minimum L2 distance to g_t. The learning process is formulated as:

$$\underset{\tilde{g}}{\text{minimize}} \ \frac{1}{2} \|g_t - \tilde{g}\|_2^2 \quad (6)$$

$$\text{subject to } \langle \tilde{g}, g_k \rangle \geq 0, \text{for all } k < t, \quad (7)$$

where $\langle \cdot, \cdot \rangle$ means the inner product. Positive inner products indicate that gradients g_k and $\tilde{g}$ are in the same direction. The updated weights by such gradients could avoid forgetting the knowledge of previous $t-1$ tasks.

2.2. Proposed Progressive Continual Learning

Inspired by progressive neural networks [24] and meta lifelong learning [25], we propose a progressive continual learning strategy for the small-footprint keyword spotting task (PCL-KWS), as shown in Figure 1.

The proposed PCL-KWS approach consists of two components: a network instantiator and a shared memory connected to a set of classification sub-networks. Specifically, we extract the mel-frequency cepstral coefficients (MFCCs) for several audio utterances in a new learning task τ_t as the inputs. Meanwhile, the proposed PCL-KWS freezes the $t-1$ sub-nets gen-

Method	ACC	LA	BWT	ACC$^+$	TT (sec)	Extra Param	Buffer Size
(a) Fine-tune (lower-bound)	$0.391_{\pm 0.265}$	$0.967_{\pm 0.015}$	$-0.264_{\pm 0.103}$	-	109.24	-	-
(b) EWC	$0.386_{\pm 0.268}$	$0.774_{\pm 0.128}$	$-0.218_{\pm 0.104}$	-0.5%	187.27	67.69K	-
(c) SI	$0.518_{\pm 0.281}$	$0.875_{\pm 0.032}$	$-0.158_{\pm 0.025}$	+12.1%	129.10	67.69K	-
(d) GEM-128	$0.622_{\pm 0.154}$	$0.939_{\pm 0.023}$	$-0.128_{\pm 0.033}$	+23.1%	291.75	-	4.10M
(e) GEM-512	$0.704_{\pm 0.117}$	$0.934_{\pm 0.019}$	$-0.115_{\pm 0.054}$	+31.3%	318.98	-	15.98M
(f) GEM-1024	$0.705_{\pm 0.109}$	$0.932_{\pm 0.020}$	$-0.101_{\pm 0.036}$	+31.4%	499.86	-	32.43M
(g) NR ($\xi = 0.5$)	$0.813_{\pm 0.007}$	$0.930_{\pm 0.004}$	$+0.001_{\pm 0.012}$	+42.2%	583.84	-	0.56G
(h) NR ($\xi = 0.75$)	$0.841_{\pm 0.011}$	$0.941_{\pm 0.010}$	$\mathbf{+0.002}_{\pm 0.001}$	+45.0%	721.49	-	0.97G
(i) PCL-KWS (fix)	$\mathbf{0.928}_{\pm 0.019}$	$\mathbf{0.949}_{\pm 0.017}$	$-0.012_{\pm 0.011}$	**+53.7%**	106.23	172.45K	-
(j) PCL-KWS	$0.884_{\pm 0.037}$	$0.944_{\pm 0.022}$	$-0.020_{\pm 0.022}$	+49.3%	**98.41**	**17.46K**	-
(k) Stand-alone (upper-bound)	$0.943_{\pm 0.003}$	$0.963_{\pm 0.247}$	$+0.002_{\pm 0.001}$	+55.2%	123.08	617.87K	-

Table 1: *The overall performance of various continual learning strategies on the TC-ResNet-8 model. PCL-KWS (fix) denotes the proposed PCL-KWS without the keyword-aware network scaling mechanism.*

erated for previous $t-1$ tasks. Next, the network instantiator generates a new t^{th} classification sub-network, which is trained with the MFCC features and metadata of the task (e.g., task id) to identify unseen keywords in the new task τ_t. The newly generated t^{th} sub-network shares the same memory along with previous sub-networks, which includes all learned knowledge. As a result, the proposed PCL-KWS can perform knowledge transfer among tasks with faster convergence speed. During inference at run-time, the proposed PCL-KWS selects the corresponding t^{th} sub-network according to the given task τ_t for evaluation.

A network instantiator is designed to generate t^{th} network for each new task τ_t. When a new keyword incremental task τ_t comes, the instantiator adds a single-head classification sub-network according to the number of the new keywords. Different from the pre-trained model including $\{16, 24, 32, 48\}$ channels, each sub-network in PCL-KWS has $\{16, 48\}$ channels. To constrain the growth of the model parameters, we propose a keyword-aware network scaling mechanism, which multiplies the channels $\{16, 48\}$ with a dynamic width multiplier α [26]. The α_τ for task τ_t is formulated as,

$$\alpha_\tau = \mu \frac{C_t}{C_0}, (\mu > 0), \tag{8}$$

where the C_t and C_0 denote the numbers of keywords in the task τ_t and task τ_0, respectively. The task τ_0 contains the data utilized to pre-train the model. In practice, the incremental task usually has less categories than the pre-trained model, leading to an $\alpha \leq 1$ that makes the network smaller.

A shared memory is designed for storing all the learned features of previous $t-1$ task, which can be accessed when learning the new task τ_t.

3. EXPERIMENTS AND RESULTS

3.1. Dataset

We conduct experiments on the *Google Speech Command* dataset (GSC) [27], which includes 64,727 one-second utterance clips with 30 English keywords categories. Following the guideline in [27], we first process all the data with a sample rate of 16kHz. We then split the dataset into two subsets, 80% for training and 20% for testing, respectively.

3.2. Experimental Setup

Testbed model. We employ the TC-ResNet-8 [28] as our testbed to explore the various continual learning strategies. It includes three residual blocks with four temporal convolution [29] layers and each layer contains $\{16,24,32,48\}$ channels. To make a fair comparison, we initialize PCL-KWS using the same settings.

Reference baselines. We built six baselines for comparisons. The *fine-tune* training strategy adapts the TC-ResNet-8 model for each new task without any continual learning strategies, as the lower-bound baseline. The *stand-alone* strategy trains the TC-ResNet-8 model for each task τ_t of the dataset separately and could obtain T separate models for evaluation. The four prior continual learning strategies (i.e., EWC, SI, GEM, NR) are introduced in Section 2. Specifically, at the training stage of naive rehearsal (NR), we replay the training samples from all previous tasks with the percentage $\xi = \{0.5, 0.75\}$. At the training stage of gradient episodic memory (GEM), we set the buffer size to $\{128, 512, 1024\}$ training samples.

CL setting. We pre-train the TC-ResNet-8 model on the GSC dataset with 15 randomly selected keywords. To evaluate the baselines along with the proposed PCL-KWS model, we simulate the incremental learning process with 5 different tasks, where each includes 3 new unique keywords. For all methods, we evaluate them on an AWS *t2.medium* instance with only 2 Intel Xeon virtual CPU cores and 4G RAM.

3.3. Evaluation Metrics

We report performance using the accuracy and efficiency metrics. The accuracy metrics include *average accuracy (ACC)*, *learning accuracy (LA)* [30], and *backward transfer (BWT)* [20]. Specifically, 1) *ACC* is the accuracy averaged on all

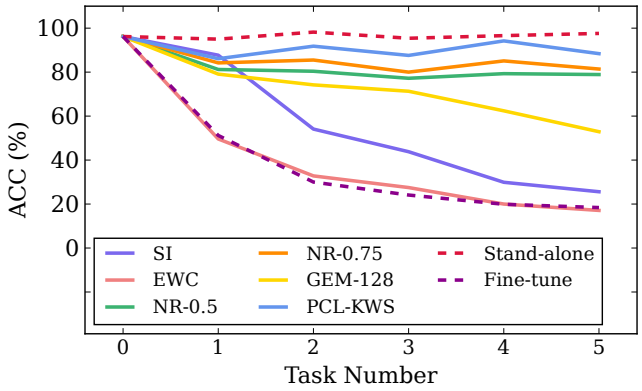

Fig. 2: *The ACC (%) with the number of learned tasks.*

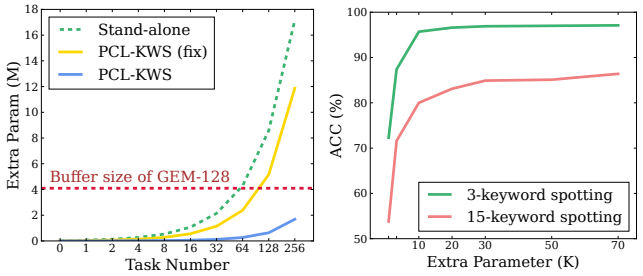

Fig. 3: *The extra parameter of PCL-KWS with the number of learned tasks (a), and the extra parameters with corresponding accuracy of the task with different keyword numbers (b).*

learned tasks, and ACC^{+} is the ACC improvement compared to the fine-tuning baseline. 2) *LA* denotes the accuracy evaluated on the current task. 3) *BWT* evaluates accuracy changes on all previous tasks after learning a new task, indicating the forgetting degree.

The efficiency metrics include *task training time (TT)*, *extra parameters (Extra Param)*, and *buffer size*. 1) *TT* is the per-epoch average training time of all the tasks. 2) *Extra Param* measures the additional parameters brought by the continual learning (CL) strategy. There are no extra parameters for reply-based methods (i.e., NR, GEM). (3) *Buffer size* indicates the extra in-memory cache occupied by the CL strategy for storing replay data or model weights.

3.4. Effect of the regularization-based methods

Table 1 shows that the EWC and SI have the worst ACCs performances compared with other baselines. Such poor performance might caused by inappropriate regularization, which restricts the ability of learning the new task when preserving prior knowledge. The sub-optimal LA performances also indicate the limited learning ability on new tasks. In addition, from the Figure 2, the performance of SI drops dramatically when the number of learning tasks increases. One reason is that the fixed model size limits the learning capacity.

3.5. Effect of the replay-based methods

From Table 1, we find that the NR strategy performs best compared with other baselines, because the data replay utilizes the historical training samples when learning new tasks. BWT shows favorable results, indicating that NR encounters no forgetting issues. Compared with the NR strategy, the GEM strategy has a much smaller buffer size. When the buffer size increases from 512 to 1024, the GEM strategy did not get the higher ACC performances but brings more heavy system workloads.

3.6. Effect of the proposed PCL-KWS

Table 1 shows that the PCL-KWS achieves the best ACC and LA performances with fewer *Extra Param* compared with other competitive methods. Specifically, compared with the best regularization-based method (i.e., SI), PCL-KWS improves the ACC by 37.2% with only 17.46K *Extra Param*. Meanwhile, compared with the best replay method (i.e., NR), the PCL-KWS reaches 8.7% ACC improvement with no buffer and 7-fold training time reduction. We then analyse the ACC performances of the proposed PCL-KWS with increasing numbers of tasks T. As shown in Figure 2, the proposed PCL-KWS always maintains a high ACC compared with the other methods, indicating good immunity to catastrophic forgetting.

We further report the *Extra Param* for the proposed PCL-KWS when there is increasing numbers of tasks T. As shown in Figure 3(a), even with $T = 256$ tasks, the *Extra Param* performances of the PCL-KWS is still less than 2M. This can be explained that the keyword-aware network scaling mechanism could significantly constrain the growth of the model parameters. In addition, we also illustrate the ACC performances of the proposed PCL-KWS with increasing *Extra Param*. As shown in Figure 3(b), the ACC performance is first improved by the increasing *Extra Param* but then he ACC curve saturates. Specifically, the proposed PCL-KWS requires at least 10K *Extra Param* for 3-keyword spotting in each new task and at least 31K *Extra Param* for 15-keyword spotting tasks.

4. CONCLUSION

In this paper, we first explore four continual learning (CL) strategies (EWC, SI, GEM, NR) for KWS keyword incremental learning. Then, we proposed a progressive CL strategy (PCL-KWS) to alleviate the problem of catastrophic forgetting. Experimental results show that the proposed PCL-KWS archives the new state-of-the-art performance of 92.8% average accuracy for all the tasks compared with other competitive baselines.

5. REFERENCES

[1] Guoguo Chen, Carolina Parada, and Georg Heigold, "Small-footprint keyword spotting using deep neural networks," in *ICASSP*. IEEE, 2014, pp. 4087–4091.

[2] Nancy F Chen, Pharri Van Tung, Haihua Xu, Xiong Xiao, Chongjia Ni, I-Fan Chen, Sunil Sivadas, Chin-Hui Lee, Eng Siong Chng, Bin Ma, et al., "Exemplar-inspired strategies for low-resource spoken keyword search in swahili," in *ICASSP*. IEEE, 2016, pp. 6040–6044.

[3] Tara N Sainath and Carolina Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[4] Nancy F Chen, Sunil Sivadas, Boon Pang Lim, Hoang Gia Ngo, Haihua Xu, Bin Ma, Haizhou Li, et al., "Strategies for vietnamese keyword search," in *ICASSP*. IEEE, 2014.

[5] Nancy F Chen, Chongjia Ni, I-Fan Chen, Sunil Sivadas, Haihua Xu, Xiong Xiao, Tze Siong Lau, Su Jun Leow, Boon Pang Lim, Cheung-Chi Leung, et al., "Low-resource keyword search strategies for tamil," in *ICASSP*. IEEE, 2015.

[6] Mark Mazumder, Colby Banbury, Josh Meyer, Pete Warden, and Vijay Janapa Reddi, "Few-shot keyword spotting in any language," *INTERSPEECH*, 2021.

[7] Abhijeet Awasthi, Kevin Kilgour, and Hassan Rom, "Teaching keyword spotters to spot new keywords with limited examples," *INTERSPEECH*, 2021.

[8] James Lin, Kevin Kilgour, Dominik Roblek, and Matthew Sharifi, "Training keyword spotters with limited and synthesized speech data," in *ICASSP*. IEEE, 2020, pp. 7474–7478.

[9] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele, "Meta-transfer learning for few-shot learning," in *Proceedings of the IEEE/CVF CVPR*, 2019, pp. 403–412.

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

[11] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[12] Yizheng Huang, Huaizheng Zhang, Yonggang Wen, Peng Sun, and Nguyen Binh Duong TA, "Modelci-e: Enabling continual learning in deep learning serving systems," *arXiv preprint arXiv:2106.03122*, 2021.

[13] Huaizheng Zhang, Meng Shen, Yizheng Huang, Yonggang Wen, Yong Luo, Guanyu Gao, and Kyle Guan, "A serverless cloud-fog platform for dnn-based video analytics with incremental learning," *arXiv preprint arXiv:2102.03012*, 2021.

[14] Heng-Jui Chang, Hung-yi Lee, and Lin-shan Lee, "Towards lifelong learning of end-to-end asr," *INTERSPEECH*, 2021.

[15] Samik Sadhu and Hynek Hermansky, "Continual learning in automatic speech recognition," in *INTERSPEECH*, 2020, pp. 1246–1250.

[16] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, and et al, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

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

[18] Yen-Chang Hsu, Yen-Cheng Liu, Anita Ramasamy, and Zsolt Kira, "Re-evaluating continual learning scenarios: A categorization and case for strong baselines," *arXiv preprint arXiv:1810.12488*, 2018.

[19] 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.

[20] 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.

[21] Ju Xu and Zhanxing Zhu, "Reinforced continual learning," *Advances in Neural Information Processing Systems*, p. 907–916, 2018.

[22] Amir Rosenfeld and John K Tsotsos, "Incremental learning through deep adaptation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 3, pp. 651–663, 2018.

[23] Nana Hou, Chenglin Xu, Joey Tianyi Zhou, Eng Siong Chng, and Haizhou Li, "Multi-task learning for end-to-end noise-robust bandwidth extension.," in *INTERSPEECH*, 2020, pp. 4069–4073.

[24] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[25] Chengyu Wang, Haojie Pan, Yuan Liu, Kehan Chen, Minghui Qiu, Wei Zhou, Jun Huang, Haiqing Chen, Wei Lin, and Deng Cai, "Mell: Large-scale extensible user intent classification for dialogue systems with meta lifelong learning," in *ACM SIGKDD*, 2021, pp. 3649–3659.

[26] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[27] Pete Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[28] Seungwoo Choi, Seokjun Seo, Beomjun Shin, Hyeongmin Byun, Martin Kersner, Beomsu Kim, Dongyoung Kim, and Sungjoo Ha, "Temporal convolution for real-time keyword spotting on mobile devices," *INTERSPEECH*, 2019.

[29] Xiang Hao, Chenglin Xu, Nana Hou, Lei Xie, Eng Siong Chng, and Haizhou Li, "Time-domain neural network approach for speech bandwidth extension," in *ICASSP*. IEEE, 2020, pp. 866–870.

[30] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro, "Learning to learn without forgetting by maximizing transfer and minimizing interference," *ICLR*, 2019.