Continual Learning for Monolingual End-to-End Automatic Speech Recognition

Steven Vander Eeckt and Hugo Van hamme

KU Leuven
Department Electrical Engineering ESAT-PSI
Kasteelpark Arenberg 10, Bus 2441, B-3001 Leuven Belgium
{*steven.vandereeckt, hugo.vanhamme*}@*esat.kuleuven.be*

Abstract—**Adapting Automatic Speech Recognition (ASR) models to new domains results in a deterioration of performance on the original domain(s), a phenomenon called Catastrophic Forgetting (CF). Even monolingual ASR models cannot be extended to new accents, dialects, topics, etc. without suffering from CF, making them unable to be continually enhanced without storing all past data. Fortunately, Continual Learning (CL) methods, which aim to enable continual adaptation while overcoming CF, can be used. In this paper, we implement an extensive number of CL methods for End-to-End ASR and test and compare their ability to extend a monolingual Hybrid CTC-Transformer model across four new tasks. We find that the best performing CL method closes the gap between the fine-tuned model (lower bound) and the model trained jointly on all tasks (upper bound) by more than 40%, while requiring access to only 0.6% of the original data.**

Index Terms—**End-to-End Automatic Speech Recognition, Continual Learning, Monolingual Speech Recognition**

I. INTRODUCTION

Automatic Speech Recognition (ASR) has greatly progressed in recent years, moving from Hidden Markov Model (HMM) to End-to-End (E2E) models. However, like the Artificial Neural Networks (ANN) they use, E2E ASR models suffer from Catastrophic Forgetting (CF) [1] when adapted to new tasks, even for monolingual tasks: it suffices that the data distributions of the old and new tasks differ for CF to occur. Fortunately, many Continual Learning (CL) methods have been proposed in the image classification community, enabling ANNs to learn continually without suffering from CF. Following [2], CL methods are categorized into three groups: i) the regularization-based methods use a regularization loss to train new tasks such that it does not hurt the performance of previous tasks. [3]–[5] do this by estimating the importance of each parameter to previous tasks and using these importance weights in a weighted L2 regularization to learn new ones. [6] uses knowledge distillation [7] on the new tasks' data to transfer knowledge from the old to the new model; ii) the replay-based methods store a set of representative samples in a memory to rehearse old tasks when learning new ones. Most straightforward are [8]–[10], which train on the new task and the memory jointly. Alternatively, [11], [12] focus on gradient alignment between old and new tasks; iii) the architectural-based methods increase the model's capacity when learning

new tasks. The latter are not considered in this paper.

Regarding ASR and, especially, E2E ASR, CL is a very new and unexplored topic. [13], [14] apply CL to the acoustic model of a HMM-based ASR model. [15] considers CL for the pre-trained wav2vec2 model [16]. [17] combines a Text-to-Speech and ASR model to prevent forgetting. [18] applies Learning without Memorizing [19] to E2E ASR, focusing on a scenario where subsequent tasks are much smaller than the initial one. Finally, [20] implements four existing CL methods for E2E ASR. Compared to [20], we implement an extra seven CL methods for E2E ASR. We test and compare their ability to continually extend and enhance a monolingual E2E ASR by training it on new data. To make the experiments more realistic, we run the methods without assuming access to validation sets of previous tasks to optimize hyper-parameters. Since for many CL methods, both regularization-based and rehearsal-based, the weight of the regularization is a crucial hyper-parameter, we propose, based on [2], a simple and efficient way to determine this weight.

II. CONTINUAL LEARNING FOR E2E ASR

We first elaborate on the considered E2E ASR model as well as on the objective of Continual Learning for E2E ASR.

Model. Our model is the Hybrid CTC/Transformer from [21]. Its loss during training is computed as:

$$\mathcal{L}(X, y; \theta) = c \cdot \mathcal{L}^c(X, y; \theta) + (1 - c) \cdot \mathcal{L}^d(X, y; \theta) \quad (1)$$

where $\mathcal{L}^c(X, y; \theta)$ and $\mathcal{L}^d(X, y; \theta)$ are, respectively, the CTC and Decoder Cross-Entropy (CE) loss of the model with parameters θ on utterance X with ground truth y. As in [21], the weight of CTC for training and decoding is $c = 0.3$. No Language Model is used during decoding. The outputs of the model are 300 word pieces, generated by the Sentence Piece model [22] on the training data of the first task.

Notation. Denote $f^c(X; \theta) \in \mathbb{R}^{L \times o}$ and $f^d(X; \theta) \in \mathbb{R}^{W \times o}$ the CTC and Decoder output, respectively, of the model with parameters θ, given utterance X, with L, W and o the utterance length, output length and number of word pieces. During training, $f^d(X; \theta)$ is conditioned on ground truth y.

Problem formulation. Let $D_1, D_2, .., D_T$ represent the labeled training datasets of the T tasks. If θ^t are the model's parameters after learning t tasks, then the objective of the CL

methods is to learn tasks $1, .., T$ in sequence such that after T tasks, θ^T, adapted from θ^{T-1}, satisfies:

$$\theta^T = \arg\min_{\theta} \sum_{t=1}^{T} \sum_{(X,y) \in D_t} \mathcal{L}(X, y; \theta) \tag{2}$$

However, when learning task T on D_T, access to $D_1, ..., D_{T-1}$ is assumed to be lost (though storing a small number of utterances per task in a memory is allowed for the rehearsal-based methods), thus θ^T cannot be directly computed from (2). In addition, we assume that we can no longer use the previous tasks' validation sets (to optimize hyper-parameters of the CL methods). We consider this a more realistic scenario.

III. CONTINUAL LEARNING METHODS

We consider both regularization- and rehearsal-based methods. As E2E ASR models have a complex architecture and are computationally demanding to train, we focus on lightweight methods which are easily applicable to any ANN architecture, and have proven to work well in other domains.

A. Regularization-based Methods

The regularization-based methods compute a regularization loss which is added to $\mathcal{L}(X, y; \theta)$ from (1) during training.
Elastic Weight Consolidation (EWC). After training task t, EWC [3] computes the diagonal of the Fisher information matrix, denoted Ω^t. Ω_{ii}^t is considered the importance weight of parameter θ_i for task t. Next, Ω^t is added to $\Omega^{\leq t} = \Omega^{\leq t-1} + \Omega^t$ as in [23], and used in the regularization loss to learn task $t+1$:

$$\mathcal{L}_{ewc}(\theta) = \frac{\lambda}{2}(\theta - \theta^t)^T \Omega^{\leq t}(\theta - \theta^t) \tag{3}$$

Since Ω^t is diagonal, (3) reduces to a weighted L2 regularization, with weight Ω_{ii}^t for parameter θ_i.
Memory-Aware Synapses (MAS). MAS [4] works similar as EWC, but computes (the diagonal of) Ω^t differently. Given that the ASR model has both a CTC and Decoder output, we compute Ω^t for MAS as follows:

$$\Omega_{ii}^t = \mathbb{E}_{X \sim D_t}\left[\left\| c\frac{\partial\|f^c(X; \theta^t)\|^2}{\partial\theta_i} + (1-c)\frac{\partial\|f^d(X; \theta^t)\|^2}{\partial\theta_i}\right\|\right] \tag{4}$$

Next, the loss is exactly the same as for EWC in (3).
Continual learning with Sampled Quasi-Newton (CSQN). CSQN [24] was proposed to extend EWC by considering interactions between parameters. Starting from EWC's Ω^t, CSQN considers quasi-Newton methods to compute low-rank approximations of the Hessian of the loss, which are then used as in (3) to regularize training. We consider both the standard and the reduced version, which was called BTREE in [24] and which we here denote CSQN-BT.
Learning Without Forgetting (LWF). LWF [6], when learning task $t+1$, uses knowledge distillation [7] between the old

model (with parameters θ^t) as teacher and the current model (with parameters θ) as student, on the new task's data:

$$\begin{aligned}\mathcal{L}_{lwf}(X; \theta) = \lambda \cdot (c \sum_{i=1}^{L} \sum_{j=1}^{o} \frac{f_{i,j}^c(X; \theta^t)}{\gamma} \log \frac{f_{i,j}^c(X; \theta)}{\gamma} \\ + (1-c) \sum_{i=1}^{W} \sum_{j=1}^{o} \frac{f_{i,j}^d(X; \theta^t)}{\gamma} \log \frac{f_{i,j}^d(X; \theta)}{\gamma})\end{aligned} \tag{5}$$

With γ called the temperature. In our experiments, $\gamma = 1$.

B. Rehearsal-based Methods

The rehearsal-based methods use a small memory of exemplars of previous tasks to enable CL.
Experience Replay (ER). We consider three variants of ER [8]. In the standard variant, the mini-batch from the current task is augmented with a mini-batch sampled from memory and sent through the model to compute the loss. As this may result in overfitting on the memory, the loss of the mini-batch sampled from memory may be given a weight $\lambda \in (0, 1)$, denoted ER (λ). Alternatively, as in [9], the training set and memory can be merged to train on the resulting set, referred to as BER (Batch-level ER).
Average-Gradient Episodic Memory (A-GEM). Consider $g = \frac{\partial\mathcal{L}(X,y;\theta)}{\partial\theta}$ with (X, y) a mini-batch from the current task. Before g is used to update the model, A-GEM [12] samples a mini-batch $(\tilde{X}, \tilde{y})$ from memory, and computes $g_{ref} = \frac{\partial\mathcal{L}(\tilde{X},\tilde{y};\theta)}{\partial\theta}$. If g and g_{ref} interfere, i.e. if $g^T g_{ref} < 0$, it updates g with $g \leftarrow g - \frac{g^T g_{ref}}{g_{ref}^T g_{ref}} g_{ref}$ such that the gradients align. The resulting gradient is used to update the model. A-GEM is the more efficient version of GEM (Gradient-Episodic Memory) [11], which was the best method in [20].
Knowledge Distillation (KD). KD uses the same loss as LWF in (5), not computed on a mini-batch of the new task, but on a mini-batch sampled from the memory. Note that this loss is added to (1), so the new task is still learned using the CE loss.

IV. EXPERIMENTS

Experiments were done in ESPnet [25]. For detailed information and more extensive results, see our repository [1].
Data. We use the Corpus Gesproken Nederlands (CGN) dataset [26], which contains 900 hours of Dutch speech from both the Netherlands (NL) and Belgium (VL). We consider all except the more spontaneous speech and, based on the dialect of the speakers, split the data into four tasks: *NL-main*, *VL-main*, *NL-rest*, *VL-rest* (learned in this order). Each task is further split into a training, validation and test set.
Training. We use the optimizer from [21], with a learning rate of 10.0 for the first task and 1.0 for subsequent tasks. We allow models to run for 230 epochs, but stop early when the Token Error Rate (TER) at word piece level on the new task's validation set has not improved for 10 epochs. As in [27], we average the last 10 snapshots to obtain a final model.

[1] https://github.com/StevenVdEeckt/CGN_CL_Dialect

Determining λ. Many of the CL methods require setting a hyper-parameter λ, the weight of the regularization. Based on [2], we propose a simple and efficient way to determine λ for E2E ASR. First, we consider τ^{init}, the TER (on the new task's validation set) of the initial model. Next, we adapt the model for five epochs without regularization, and compute its TER, obtaining τ^{no_reg}. Then, we set λ to a high value and run the model for five epochs with regularization with weight λ. We compute the TER and obtain τ. If $(\tau - \tau^{init})/(\tau^{no\_reg} - \tau^{init}) > a$, i.e. if the gap between τ^{init} and τ^{no_reg} is closed for at least $100a\%$, we return λ; else, we set $\lambda \leftarrow p\lambda$ with $p \in (0,1)$ and repeat the process. As such, determining λ is done in a fast and efficient way and does not require access to a validation set of previous tasks. We determine λ only for the first adaptation and then fix it. In our experiments, we set $a = 0.85$ and $p = 0.10$. Moreover, for each method, the initial value of λ is a power of 10.

Memory. After learning a task, we sample 500 utterances from the training set to add to the memory. While sampling uniformly, we only consider utterances whose output length (i.e. number of word pieces in output) exceeds $0.40 \cdot mean\_length$, where $mean\_length$ is the average output length of the utterances in the training set, to make sure that all of the 500 utterances contain meaningful sentences.

Baselines. Following baselines are considered: (i) Fine-Tuning (FT): the model is adapted without CL method (lower bound); (ii) Joint (JT): trained from scratch on all tasks jointly; (iii) Continued Joint (CJT): adapted from previous task and trained on current and previous tasks jointly (upper bound).

Metrics. For each method, we report the Average WER (AWER), Backward Transfer (BWT) and Forward Transfer (FWT) [11], and Coverage (COV) [13]. Assuming T tasks have been learned and $R_{i,j}$ is the WER on task j after learning up to task i, $AWER = \sum_{i=1}^{T} R_{T,i}$, while the BWT is:

$$\text{BWT} = \frac{1}{T-1} \sum_{i=1}^{T-1} -(R_{T,i} - R_{i,i}) \quad (6)$$

Note that using this definition, negative BWT indicates forgetting. Furthermore, we define FWT as:

$$\text{FWT} = \frac{1}{T-1} \sum_{i=2}^{T} -(R_{i,i} - R_{i,i}^{FT}) \quad (7)$$

where $R_{i,j}^{FT}$ is the WER on task j after learning up to task i with FT. FWT measures to which extent the model can exploit previously acquired knowledge to learn new tasks better. Positive FWT indicates better learning than FT. Finally, COV measures the extent to which the given method closes the gap between FT (lower bound) and CJT (upper bound) in terms of AWER. It is 0% when the method performs as poor as FT, and 100% when the method performs as well as CJT. In addition to AWER, BWT, FWT and COV, we report the storage requirements (Storage), expressed in an equivalent number of models (one model requiring 105 MB).

Statistical significance. We use the Wilcoxon signed-rank test on the number of errors per utterance [28] to test the

TABLE I: Results after learning the four tasks. Significance level refers to improvement over baseline FT.

Model	AWER↓	BWT↑	FWT↑	COV↑	Storage
JT	21.3	-	-	-	260.54
CJT	21.9	+2.5	+0.5	100.0%	261.54
FT	27.3	-4.2	-	0.0%	1.00
EWC	28.3	**-0.7**	-4.8	-18.9%	2.00
MAS	28.3	-1.1	-4.4	-18.9%	2.00
CSQN	27.7	-1.5	-3.2	-8.7%	32.00
CSQN-BT	27.8	-1.7	-3.2	-9.8%	22.00
LWF	26.6***	-3.3	**+0.1**	12.4%	1.00
A-GEM	26.1***	-2.5	-0.0	22.0%	3.24
ER	28.0	-3.4	-1.7	-13.1%	3.24
ER (λ)	25.8***	-1.9	-0.3	27.2%	3.24
BER	26.4***	-2.8	-0.2	16.7%	3.24
KD	**25.0*****	-1.2	+0.0	**41.7%**	3.24

TABLE II: WER on Test set and Memory of initial task *NL-main* after the first adaptation to *VL-main*. 'Initial' is the model trained on *NL-main*, from which the other models are adapted.

Model	Test	Memory
Initial	27.1	10.7
FT	33.0	21.8
A-GEM	31.0	2.6
ER	32.7	0.0
ER (λ)	30.6	0.4
BER	32.3	13.9
KD	29.4	10.5

significance of the results, considering significance levels $\alpha = 0.05$ (*), $\alpha = 0.01$ (**) and $\alpha = 0.001$ (***).

V. RESULTS

Table I shows the results after learning the four tasks in sequence. First, we note that FT indeed suffers from CF, while both JT and CJT are able learn the tasks well, with the latter reaching a positive BWT and FWT.

Considering the regularization-based methods, we find that the methods estimating which parameters are important experience difficulties learning the four tasks. This is especially true for EWC and MAS, both performing worse than FT. While CSQN and CSQN-BT, by considering interactions between parameters, perform slightly better, they still underperform FT. We hypothesize that the poor performance of these methods is due to the tasks (being very similar) having the same important parameters, which gives the model two options: either it updates these parameters, resulting in CF of previous tasks; or it leaves them unchanged, resulting in poor learning of the new tasks. In this experiment, EWC, MAS and CSQN reduce forgetting, but fail to learn the new tasks well. While EWC achieves the best BWT of all methods, it also attains the worst FWT. Note that the performance of EWC is in line with [20], which also found EWC underperforming FT. Compared to EWC, MAS and CSQN, LWF performs much (and significantly, with $\alpha = 0.001$) better, achieving the highest FWT (higher than FT). However, its COV is only 12.4%, as it reduces FT's forgetting (BWT) by only 21%.
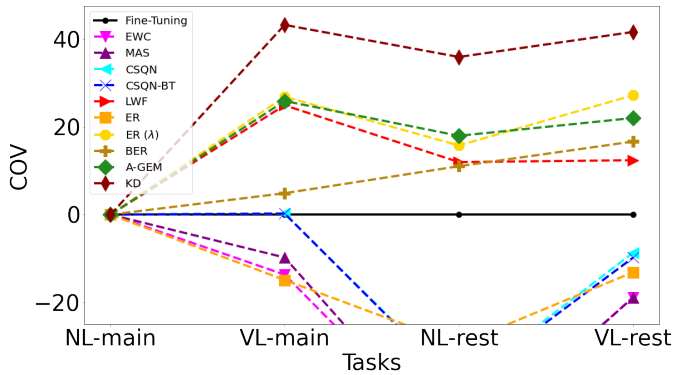
Fig. 1: COV after learning the four tasks.

TABLE III: Results after learning the four tasks with fixed memory. Significance level refers to deterioration compared to corresponding method from Table I (with increasing memory).

Model	AWER↓	BWT↑	FWT↑	COV↑	Storage
A-GEM	26.2*	-2.8	-0.0	18.9%	1.72
ER (λ)	25.6	-1.6	-0.4	29.8%	1.72
KD	**25.2***	**-1.5**	**+0.1**	**38.3%**	1.72

also benefit *NL-main*, while protecting *VL-main*. While this is a realistic scenario when extending a monolingual ASR model, it turns out to be a very challenging one, especially for EWC, MAS and CSQN, which, by protecting *NL-main*'s important parameters, are unable to exploit *NL-rest* to further improve these parameters.

A. Increasing vs. Fixed Memory

The rehearsal-based methods from Table I had access to a memory with 500 utterances per task. In practice, it may be more desirable and/or feasible to have a memory with fixed size, especially as the number of tasks becomes large. To this end, we fix the size of the memory at 500. Table III shows the results for A-GEM, ER (λ) and KD.

We find that the differences with Table I are small. For A-GEM and KD, we observe a slight (though significant, with $\alpha = 0.05$) deterioration of AWER. For ER (λ), on the other hand, there is even a minor improvement, which, not being statistically significant, is attributed due to chance. Even for a very small and fixed memory (with only 0.2% of original training data after four tasks), A-GEM and, in particular, ER (λ) and KD are thus highly effective in enabling CL.

B. Storage Requirements

Table I shows the storage requirements of the CL methods to learn the fourth task. For all methods except JT, which for each new task starts from scratch, this requires, first of all, the storage of the model itself. In addition, the rehearsal-based methods require storing an equivalent of 2.24 models, as they need to store the utterances in the memory.

Compared to the rehearsal-based methods, the regularization-based methods are more storage efficient (in addition to not requiring data from previous tasks to be stored in a memory, which may be, due to e.g. privacy concerns, not always allowed). While LWF requires storing only the previous model, EWC and MAS, in addition, need to store the importance weights. Compared to the latter, CSQN and CSQN-BT are less storage efficient, due to the Hessian approximations.

With an increasing memory, as in Table I, the rehearsal-based methods' storage requirements also increase linearly with the number of tasks. However, as we saw in Table III, this can be overcome by fixing the memory size, with only a negligible deterioration in performance, enabling A-GEM, and, especially, ER (λ) and KD, to achieve excellent performance while being very storage efficient, requiring to store an equivalent of only 1.72 models (independent of number of tasks). Finally, note how JT and CJT, needing access to all data the model

Comparing LWF to KD, which uses the same regularization but computed on the memory instead of on the new task's data, we find that having access to a memory, even though it is only 0.6% of the original data when learning the fourth task, yields big improvements (with significance level $\alpha = 0.001$). KD attains a COV of over 40%, and learns the new tasks as well as FT, while reducing the latter's forgetting by more than 70%. It outperforms the other rehearsal-based methods by a large margin (with significance level $\alpha = 0.001$). A-GEM, while it learns the new tasks well, still suffers from severe forgetting, reaching a COV of 22%. This is again consistent with [20], which found GEM (of which A-GEM is a more efficient variant) outperforming LWF, while both improved the performance of FT. Finally, ER performs worse than FT, as it suffers from CF and is unable to learn the new task well. Both BER, and especially ER (λ), perform much better, reaching a COV of 16.7% and 27.2%, respectively.

Table II gives us more insight into how the rehearsal-based methods work. It shows the WER on the memory and test set of the initial task *NL-main* of models adapted to *VL-main*. For ER, we note that it memorizes the memory completely, achieving 0.0 WER, and this generalizes very poorly to the test set. ER (λ) alleviates this, though it still almost perfectly memorizes the memory set. A-GEM, too, has a very low WER on the memory set. On the other hand, KD only very slightly improves on the memory set, but it is able to extract much more 'general' knowledge from it, limiting the forgetting on the test set much better than ER, ER (λ) or A-GEM.

Figure 1 shows the COV after learning each task. We find that after two tasks, LWF performs as well as A-GEM and ER (λ). However, as more tasks are added, the gap between LWF, and A-GEM and ER (λ) widens. Moreover, while BER after two tasks only slightly outperforms FT, its performance, relative to FT and the other CL methods, improves as more tasks are added. This is as expected, since BER, learning on the merger of the training set and the memory, clearly benefits from having a larger memory. Finally, note how many CL methods' performance drops when learning *NL-rest*. That is because *NL-main* and *NL-rest* are very similar, so the CL methods should enable the model to learn *NL-rest*, as this will

was ever trained on, require storing an equivalent of 260.54 and 261.54 models, respectively, making them clearly not a practical solution to overcome CF.

VI. CONCLUSION

In this paper, we implemented an extensive number of CL methods, and tested and compared their ability to extend a monolingual E2E ASR model across four tasks. Having access to a memory, though very small compared to the original training set, proved to be very beneficial, as the rehearsal-based methods generally performed much better than the regularization-based methods. To assure the former's storage requirements do not increase with the number of tasks, the memory size can be fixed with only a negligible degradation in performance. In general, thus, the rehearsal-based methods seem the best and most practical way to currently overcome CF in monolingual E2E ASR models; in particular KD, which closes the gap between the Fine-Tuned (lower bound) and Continued Joint model (upper bound) for 41.7% and 38.3% while having access to only 0.6% and 0.2%, respectively, of the original data. In case storing utterances from previous tasks is not allowed, LWF seems to be the best option, as the other regularization-based methods, which have higher storage requirements, were unable to improve the Fine-Tuning lower bound. However, even in case only a very small number of utterances per task can be stored, it is advised to do so.

REFERENCES

[1] Michael McCloskey and Neal J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," vol. 24 of *Psychology of Learning and Motivation*, pp. 109–165. Academic Press, 1989.

[2] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1–1, 2021.

[3] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[4] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Computer Vision – ECCV 2018*, Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, Eds., Cham, 2018, pp. 144–161, Springer International Publishing.

[5] Friedemann Zenke, Ben Poole, and Surya Ganguli, "Continual learning through synaptic intelligence," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. 2017, ICML'17, p. 3987–3995, JMLR.org.

[6] Zhizhong Li and Derek Hoiem, "Learning without forgetting," in *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, Eds., Cham, 2016, pp. 614–629, Springer International Publishing.

[7] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[8] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne, "Experience replay for continual learning," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. 2019, vol. 32, Curran Associates, Inc.

[9] Zheda Mai, Hyunwoo J. Kim, Jihwan Jeong, and Scott Sanner, "Batch-level experience replay with review for continual learning," *ArXiv*, vol. abs/2007.05683, 2020.

[10] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet Kumar Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato, "On tiny episodic memories in continual learning," *arXiv: Learning*, 2019.

[11] David Lopez-Paz and Marc'Aurelio Ranzato, "Gradient episodic memory for continual learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2017, NIPS'17, p. 6470–6479, Curran Associates Inc.

[12] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny, "Efficient lifelong learning with a-gem," in *ICLR*, 2019.

[13] Brady Houston and Katrin Kirchhoff, "Continual Learning for Multi-Dialect Acoustic Models," in *Proc. Interspeech 2020*, 2020, pp. 576–580.

[14] Samik Sadhu and Hynek Hermansky, "Continual Learning in Automatic Speech Recognition," in *Proc. Interspeech 2020*, 2020, pp. 1246–1250.

[15] Samuel Kessler, Bethan Thomas, and Salah Karout, "Continual-wav2vec2: an application of continual learning for self-supervised automatic speech recognition," *arXiv*, 2021.

[16] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds. 2020, vol. 33, pp. 12449–12460, Curran Associates, Inc.

[17] Amin Fazel, Wei Yang, Yulan Liu, Roberto Barra-Chicote, Yixiong Meng, Roland Maas, and Jasha Droppo, "Synthasr: Unlocking synthetic data for speech recognition," *CoRR*, vol. abs/2106.07803, 2021.

[18] Li Fu, Xiaoxiao Li, and Libo Zi, "Incremental learning for end-to-end automatic speech recognition," *ArXiv*, vol. abs/2005.04288, 2020.

[19] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa, "Learning without memorizing," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5133–5141, 2019.

[20] Heng-Jui Chang, Hung yi Lee, and Lin shan Lee, "Towards Lifelong Learning of End-to-End ASR," in *Proc. Interspeech 2021*, 2021, pp. 2551–2555.

[21] Shigeki Karita, Nelson Enrique Yalta Soplin, Shinji Watanabe, Marc Delcroix, Atsunori Ogawa, and Tomohiro Nakatani, "Improving Transformer-Based End-to-End Speech Recognition with Connectionist Temporal Classification and Language Model Integration," in *Proc. Interspeech 2019*, 2019, pp. 1408–1412.

[22] Taku Kudo and John Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Brussels, Belgium, Nov. 2018, pp. 66–71, Association for Computational Linguistics.

[23] Ferenc Huszár, "On quadratic penalties in elastic weight consolidation," *ArXiv*, vol. abs/1712.03847, 2017.

[24] Steven Vander Eeckt and Hugo Van hamme, "Continual learning with quasi-newton methods," *TechRxiv*, Sep 2021.

[25] Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai, "ESPnet: End-to-end speech processing toolkit," in *Proceedings of Interspeech*, 2018, pp. 2207–2211.

[26] Nelleke Oostdijk, "The spoken dutch corpus: Overview and first evaluation," *Proceedings of LREC-2000, Athens*, vol. 2, 01 2000.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, L ukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[28] Helmer Strik, Catia Cucchiarini, and Judith M. Kessens, "Comparing the recognition performance of csrs: in search of an adequate metric and statistical significance test," in *INTERSPEECH*, 2000.