Continual Learning in Medical Imaging: CS7643 Final Report

Amritpal Singh[1], Mustafa Burak Gurbuz[1], Shiva Souhith Gantha[1], and Prahlad Jasti[1]

[1] College of Computing , Georgia Institute of Technology

***Abstract*—In the clinical setting, Deep Learning models must be able to adapt to dynamic changes in the data, particularly in medical imaging, where new devices, protocols, patient populations, and diseases may emerge. However, the performance of these models may significantly decrease when exposed to new data during deployment, resulting in critical robustness issues. Therefore, it is crucial to continually train the models as the data changes. In this study, we evaluate the effectiveness of various state-of-the-art Continual Learning approaches on a medical imaging dataset to assess their potential for improving robustness and compare their performance. We implement and test 4 Continual Learning models namely Neuro-Inspired Stability-Plasticity Adaptation (NISPA), Replay using Memory Indexing (REMIND), Dark Experience Replay (DER) and Memory Aware Synapses (MAS). We show that a single model can sequentially learn to detect more and more pathologies in different medical imaging modalities (e.g., CT scan, Ultrasound, and X-ray).**

Github code https://github.com/BurakGurbuz97/continual_learning_medical_images

I. Introduction

Deep Neural Networks (DNNs) have recently exhibited remarkable achievements across a variety of tasks, surpassing human expertise in some instances [8], [14], [24]. However, their dependence on fixed, balanced, and shuffled datasets within stable environments presents a significant constraint. The ever-changing nature of the real world necessitates networks capable of sequential learning over time, adapting to shifting data distributions—a crucial aspect for many applications.

This shortcoming is especially pronounced in the domain of medical imaging. Consider a DNN designed to analyze patients' CT scans, which needs to be updated to identify novel biomarkers associated with emerging diseases. Fine-tuning the model exclusively on these new biomarkers results in a rapid loss of previously acquired knowledge, with the model solely adapting to the latest targets. This formidable challenge, known as Catastrophic Forgetting (CF), is tackled by the field of Continual Learning (CL), striving to develop effective solutions to this complex issue.

CL is an area of deep learning that focuses on relaxing the assumption that all relevant training data is available to a model at once. Like the use case explored in this paper, models will have to be exposed to external changes in their environment. Thus, CL algorithms train models on various scenarios, or series of data streams that are categorized in different ways. These algorithms measure the effect of CF by evaluating the validation accuracies of previous data streams within the scenario, along with the currently measured one, to ensure that the validation accuracies from previous streams do not plummet when evaluated in the current context.

As an example, consider the MNIST [7] dataset of handwritten digits from 0 to 9. We can group the examples of each class together such that the images of 0 and 1 are together, the images of 2 and 3 are together, etc. Each of these 5 groups is considered a context. While in a context, it is possible that we know the two choices of digits that the current example could be, and the model picks one or the other. This type of CL is known as task incremenetal learning (TIL), as it is known that the task is different from previous contexts. However, it is also possible that these boundaries between contexts are not known to the model, so it now has to pick from all 10 digits. This type of CL is known as class incremental learning (CIL). In this paper, we will be performing CIL, as it is the type of learning that more realistically occurs in the CL space and is more vulnerable to CF.

CL algorithms can fall into three different categories, although they are not mutually exclusive:

1) **Regularization:** Important weights relevant to learning previous tasks are identified via regularization, similar to how L1/L2 regularization identifies weights that may cause overfitting [3], [6], [13], [15], [23]. For example, Memory Aware Synapses [3], which is one of the approaches explored in this paper, assign importance for parameters on every task and penalize their change proportional to their importance.
2) **Architectural:** The learner can dynamically allocate new neurons or new parameters for specific tasks [4], [16], [20]–[22], [28]. For example, Progressive Neural Networks [21] add sub-networks and perform knowledge transfer to these sub-networks via lateral connections, retaining knowledge from previous tasks while also able to learn new tasks.
3) **Replay:** Few samples (e.g., as small as 100) from previous tasks are stored and replayed in future tasks to help the model remember these previous tasks [11], [18], [19]. While this approach may seem trivial to implement, it also detrimentally affects resource optimization due to the added training time from replaying samples.
   Replay is also the only way to achieve satisfactory CIL performance [25]. This is due to the fact that archi-

tectural approaches primarily depend on task-specific information to designate particular layers or units for specific tasks. These methods require a mechanism that can supply task information, even for test samples. In contrast, regularization techniques struggle to differentiate between classes that are not encountered concurrently in the same learning episode or task. For example, a common loss function, such as Cross Entropy, enforces the learning of decision boundaries that accurately classify given classes. Nonetheless, when faced with a novel class (a frequent occurrence in CL), output units trained using Cross Entropy tend to behave unpredictably, often producing overconfident and arbitrary predictions for new classes. This issue is closely associated with a well-known challenge referred to as the calibration problem in DL models.

When prior data remains accessible and computational resources are not a constraint, the typical strategy is to train a new model from scratch using a combination of existing datasets and new targets. In this project, we test existing CL algorithms in the medical imaging domain. Our goal is to avoid training from scratch and retain only a tiny fraction of previous data while preserving past knowledge and adapting to new information.

If we successfully develop and test effective Continual Learning (CL) algorithms in the medical imaging domain, it will make a significant difference in at least three ways:

1) **Efficient adaptation to new information:** DNNs that quickly adapt to novel biomarkers and emerging diseases without losing prior knowledge will enhance the diagnostic process's efficiency and accuracy in real-world scenarios.
2) **Resource optimization:** CL models require minimal computational resources and data, making them more accessible and sustainable for healthcare providers and institutions.
3) **Improved patient care:** CL algorithms will enable healthcare professionals to diagnose and treat patients more effectively by rapidly identifying and analyzing novel medical imaging biomarkers.

We evaluated various CL techniques on two data streams generated using MedMNIST [27]. The first stream assesses the capacity of our models to identify pathologies in microscopic images, comprising three learning episodes with each introducing 8 or 9 new targets. The second stream evaluates the models' ability to detect pathologies in radiological images, consisting of six learning episodes, in which the initial episode introduces 5 new classes and the subsequent episodes introduce 2 new classes each. Please see **Figure** 1 and **Figure** 2.

II. Approach

We implemented four CL methods, which are explained in the following subsections. Furthermore, we trained a Naive Learner (NAIVE), which is trained sequentially on stream without any specific mechanism to preserve knowledge from previous tasks. Of course, NAIVE suffers significant forgetting and is useless in practice, but it represents the lower bound performance.

A. Neuro-inspired Stability-Plasticity Adaptation

Neuro-Inspired Stability-Plasticity Adaptation (NISPA) [9] categorizes all units, such as filters in convolutional layers or neurons in linear layers, into three groups: plastic, candidate stable, and stable units. Plastic units are designated for future learning, candidate stable units are assigned to learn currently available classes, and stable units maintain knowledge about previously learned targets and are not changed.

NISPA begins with a sparsely connected DNN, which is pruned to a fixed density (30% in our experiments) through random pruning. It then employs a sparse training method with frequent connection rewiring to generate new input-output paths, facilitating the learning of new classes while leveraging existing knowledge contained in stable units. This rewiring mechanism is inspired by the structural plasticity of biological neurons and driven by local activations of units resembling Hebbian Learning in the brain.

Besides dynamic connectivity driven by activations and freezing stable units to tackle forgetting, NISPA also augments training batches by concatenating few examples from past classes to improve the consolidation of past knowledge while integrating new knowledge. Moreover, the original NISPA rewires connectivity until it starts hurting the performance of the current task (measured on a hold-out validation dataset). Instead, in this work, we fixed the number of rewiring events which removes the necessity of having a hold-out validation dataset.

B. Replay using Memory Indexing

REMIND [10] is a novel brain-inspired method for training the parameters of a CNN using replay. It can be implemented in streaming or batch training settings. To have a fair comparison with all other methods, we implemented the batch training version, which was shown to perform better than streaming in the original paper.

The method involves compressing the current input and reconstructing a subset of previously compressed representations mixed with the current input to update the plastic weights of the network. REMIND stores compressed mid-level CNN features using Vector quantization[1], which allows for far more instances to be stored with a smaller memory budget. The method is composed of two nested functions: G (·) and F (·), where G (·) is fixed, and F (·) is trained. The output is quantized and reconstructed during inference before being passed to F (·).

It is important to mention that REMIND stores feature representations instead of the actual input data, which, in our context, involves sensitive patient information. This characteristic makes REMIND particularly well-suited for situations where retaining past data indefinitely is not feasible.

[1]Following the paper, we used Product Quantization which splits given data into smaller slices and uses K-means clustering to find centroids that best describe the data.

C. Dark Experience Replay

Dark Experience Replay (DER) [5] is a replay method that makes a trade-off between replaying past experiences and ensuring an efficient and stable training process by selectively choosing examples with high uncertainty to replay. It involves storing experiences that are deemed "dark" or anomalous, meaning they have a high prediction error or low probability under the current policy. These examples are stored in a buffer during the first time if their cross entropy varies from the expected output by a certain threshold, and they are sampled uniformly from this buffer in future tasks. These examples make another forward pass across the current model, and their losses compared to the previously calculated logits are added to the loss of the current batch, after multiplying this added loss by a regularization constant α. Because these experiences are now replayed on a model with training experience from multiple tasks, it helps prevent over-fitting and improve exploration.

Dark Experience Replay++ (DER++) is a variant of DER that incorporates meta-learning, which is the process of observing the training process of a model in order to see how to learn examples more efficiently. DER++ introduces a meta-learner that predicts the expected change in the model's performance for each stored experience, and it uses these predictions to dynamically adjust the replay priority of each experience. During the training process, after performing the first replay in vanilla DER, it samples another example and performs a similar loss calculation with respect to output labels, as opposed to logits, before multiplying this loss by a constant β and adding it to the current loss. The meta-learner determines which points to sample based on their first replay performance, allowing for a more granular approach to the optimization process.

D. Memory Aware Synapses

Memory Aware Synapses (MAS) [3] is a regularization method that calculates importance of the model parameters in an online fashion. This approach is inspired from the Hebbian Learning model for the brain. MAS involves calculation of the an importance value for each parameter, which is called as Omega (Ω). It then penalizes changes to the parameters proportional to their importance when learning a new task. This penalty in effect helps the model from forgetting the important knowledge from previous tasks. The importance weights are calculated by the sensitivity of the learned function to a parameter change. It can be formulated as:

$$\Omega_{ij} = \frac{1}{N} \sum_{k=1}^{N} \|g_{ij}(x_k)\| \tag{1}$$

where x_k is a data point, N is the number of data points we average over; $\|g_{ij}(x_k)\|$ is the norm of the gradient of the output with respect to parameter θ_{ij} and Ω_{ij} is the importance of the parameter θ_{ij}.

We update the Ω after learning each task, also we store the parameters. And while learning a new task, we use the Ω values to add a regularization term to the loss function that penalizes the changes in the parameter values weighted with their importance as shown in the next equation.

$$L(\theta) = L_n(\theta) + \lambda \sum_{ij} \Omega_{ij} (\theta_{ij} - \theta^*_{ij})^2 \tag{2}$$

where L_n is the loss for the new task, λ is a hyperparameter which determines the amount of regularization loss to be added, θ^*_{ij} are the stored parameters after learning the previous task.

We also implement a variation of MAS by integrating Replay. MAS works best for the setting of TIL, therefore to improve its performance on the scenario of CIL, we integrate replay with it. In this method, we store past experiences in a memory buffer and replay them by randomly sampling past memories while learning a new task.

III. Experimental Setting

A. Dataset and Scenarios

We create two continual learning scenarios: One for microscopic images and another for radiological images.

1) **Microscopic scenario**: A microscope is a commonly used tool to visualize cellular and tissue structure, across different kinds of investigations. We created three subtasks of microscopic image datasets: Tissue images(PathMNIST [12]), Blood cells(BloodMNIST [1]) and kidney cortex cells(TissueMNIST [17]). The PathMNIST dataset was derived from a previous research study that aimed to predict survival rates based on colorectal cancer histology slides with hematoxylin and eosin-stained histological images. The dataset comprises nine different types of tissues, making it a multi-class classification task. The BloodMNIST dataset consists of images of normal individual cells that were captured from people without any infections, blood or cancer diseases.The dataset comprises a total of 17,092 images and is categorized into 8 classes. The TissueMNIST dataset comprises of 236,386 human kidney cortex cells that have been segmented from three reference tissue samples and are classified into eight categories. Figure 1 shows samples from each dataset,
2) **Radiological scenario**: Another common investigation in a healthcare setting involves radiological images. We create three subtasks of radiological image datasets: CT scans (OrganCMNIST [26]), Ultrasound (BreastMNIST [2]) and Xray(PneumoniaMNIST [12]). The OrganCMNIST dataset contains a coronal view of 3D computed tomography (CT) images obtained from the Liver Tumor Segmentation Benchmark (LiTS) dataset. The PneumoniaMNIST dataset consists of 5,856 pediatric chest X-ray images and is designed for binary-class classification of pneumonia versus normal cases. The BreastMNIST dataset comprises 780 low-resolution breast ultrasound images as a binary classification problem, where normal

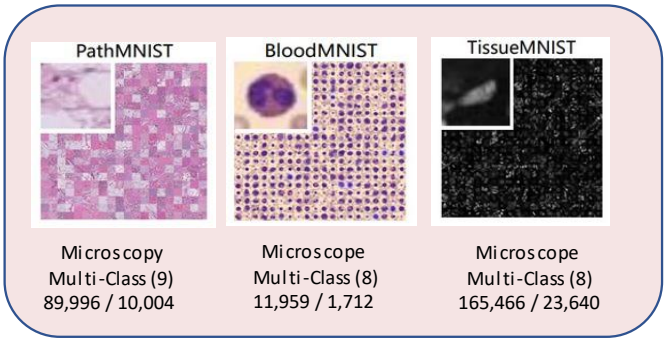

Figure 1: Microscopic data scenario

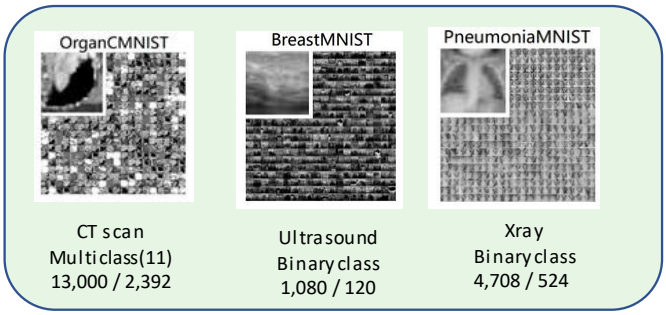

Figure 2: Radiological data scenario

and benign cases are labelled as positive against malignant cases as negative.

We use MEDMNIST [27] to access preprocessed datasets with images of size 28*28*3 resized.

B. Backbone Architecture

We use a simple convolutional neural network(CNN) as backbone model. Listing 1 shows the architecture used. In general, Conv_2d layers represents a 2d convolutional layer (nn.Conv2d) and linear_layer represents a full connected dense layer (nn.linear). In the case of NISPA, we implemented custom Pytorch classes to simulate sparsely connected layers. All convolutional layers have a padding of 0 and a stride of 1, so dimension reduction is only made in pooling layers.

Listing 1: CNN architecture

```
[1]  conv2d_layer(in_channels=3,
     out_channels = 64, kernel_size =3)
[2]  nn.ReLU()
[3]  conv2d_layer(64, 64, 3)
[4]  nn.ReLU()
[5]  nn.MaxPool2d(2)

[6]  conv2d_layer(64, 128, 3)
[7]  nn.ReLU()
[8]  conv2d_layer(128, 128, 3)
[9]  nn.ReLU()
[10] nn.MaxPool2d(2)
[11] linear_layer(128*8*8, 2000)
[12] nn.ReLU()
[13] linear_layer(2000, num_classes)
```

For REMIND, we use the first 5 layers as the feature extractor and 6-13 as the classifier part. During the first task, both parts are trained. At the beginning of the second task, the first 5 layers are frozen, and the rest is kept plastic and trained on sequential episodes.

C. Method hyperparameters

Table I shows the hyper-params used in different methods.

D. Metrics

In the context of continual learning, it is customary to evaluate models after each learning episode, which we adopt in our study. Let $A_{i,j}$ represent the accuracy on classes learned during episode (or task) i, assessed after completing episode j. We report two metrics:

- Accuracy on episode i after learning the final episode f, denoted as $A_{i,f} \quad \forall i \leq f$. This metric quantifies the performance of the final continual learning model across all classes and episodes. See Figure 5 for an example.
- Average accuracy on seen classes after completing episode j (denoted as AA_j). Formally, we compute $AA_j = \frac{1}{j}\sum_{i=1}^{j} A_{i,j}$. This metric measures the model's performance throughout the learning trajectory. It is important to note that a monotonically decreasing curve is expected when plotting AA_j against increasing j. This is because, as j grows, the number of target classes also expands, rendering classification more challenging. See Figure 3 for an example.

IV. Results and Discussion

We train the small CNN model described in section III-B with each of the approaches mentioned in section II. In this section we present and analyse the results.

A. Performance Comparison:

Figures 3 and 5 show the results on Microscopic scenario. This is a simpler scenario since we only have three sequential learning episodes. We observe that NISPA and REMIND perform nearly on par with each other, demonstrating a significant advantage over MAS Replay, DER, and DER++ in terms of maintaining accuracy on previous tasks. It is evident that MAS Replay, DER, and DER++ exhibit partial forgetting of Task-1 and Task-2, resulting in inferior performance when compared to NISPA and REMIND (refer to Figure 5).

On the other hand, Figures 4 and 6 show accuracies on a more challenging scenario with 6 sequential learning episodes. In this instance, REMIND lags behind NISPA in terms of performance. Additionally, DER and DER++ manage to close the performance gap with REMIND. Recall that REMIND relies on frozen feature extraction layers. We posit that, in this scenario, the first task lacks sufficient diversity as a pretraining dataset, causing REMIND's frozen features to struggle with

Method	Memory buffer/class	batch size ratio	Epochs	HyperParam
Naive Learner	N/A	N/A	3	lr: 1.0, Optimizer: Adadelta
NISPA	100	1	N/A	phase epochs: 5, lr: 1.0, Optimizer: Adadelta, # phases 20, density: 30%
REMIND	100	1	10	lr = 0.0005, Optimizer: SGD codebooks:32, codebook size:256
DER	100	1	10	α: 0.75, Optimizer: Adam, lr: 0.001
DER++	100	0.5	10	α: 0.75, β: 0.5, Optimizer: Adam, lr: 0.001
MAS	N/A	1	5	$\lambda = 10$, Optimizer: SGD, lr = 0.001
MAS Replay	100	1	5	$\lambda = 1$, Optimizer: SGD, lr = 0.001

Table I: Methods details: backbone architecture and hyperparameters. Memory buffer/class refers to the total number of replay images stored per class. The batch size ratio refers to ratio of new scans w.r.t replay scans from memory. Batch size is 256 for all methods. NISPA trained with Adadelta by default. It is a sparse network, and plain SGD leads to unstable training. For Naive, the optimizer does not matter since it always forgets all past classes. That's why we just kept it the same with NISPA.

generalizing to unseen examples. On the other hand, NISPA again outperforms others, demonstrating robust performance and mitigating forgetting. These findings indicate that hybrid approaches like NISPA, which combine architectural concepts (e.g., dynamic connectivity) with replay techniques (e.g., augmented replay batches), exhibit greater resilience in dynamic environments.

B. Regularization methods fails in CIL:

As we mentioned before, regularization methods suffer in the CIL setting. We see that MAS, which is a regularization method is not able to remember the previous tasks as shown in Figure 3 and Figure 4. The accuracy of MAS is similar to that of the NAIVE implementation. Although MAS fails in the CIL setting, when we provide it with the task IDs, i.e., in the TIL setting, it performs well with an average accuracy of 41.67% for the Microscopic scenario and 47.66% for the Radiological scenario respectively. In order to improve the performance of MAS in the CIL setting, incorporate replay with MAS, wherein we randomly samples examples from previous tasks while learning a new task. This helps the model remember previous tasks much better as seen in the Figures 3 and 4.

C. REMIND addresses data privacy concerns:

REMIND stores compressed representations, instead of the actual data. This allows two key advantages: memory efficiency and data privacy. Data privacy is an important aspect of the healthcare setting. Using models like REMIND can allow models to be trained inside a hospital setting, without sharing real data. Another key point to note is REMIND's dependance on its feature extractor. Since the inital feature extractor is frozen after its initialization, this makes model less flexible to learning tasks that are unrelated to initial tasks. We train feature extractor on the first task to train the feature extractor. Using pre-trained weights from bigger datasets can provide a boost in model performance.

V. Code Availability

Code is available at Github repo: https://github.com/ BurakGurbuz97/continual_learning_medical_images

VI. Work Division

Table II shows the work distribution among the team members.

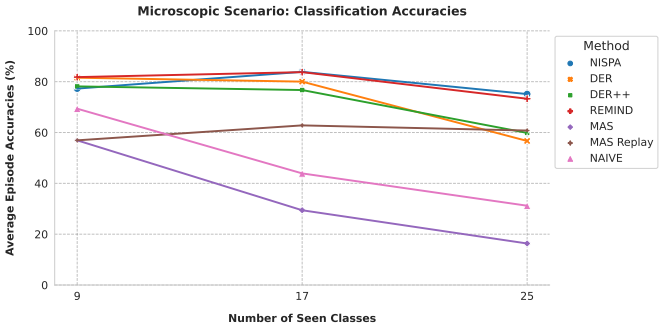

Figure 3: Average accuracy vs tasks seen: Microscopic data scenario

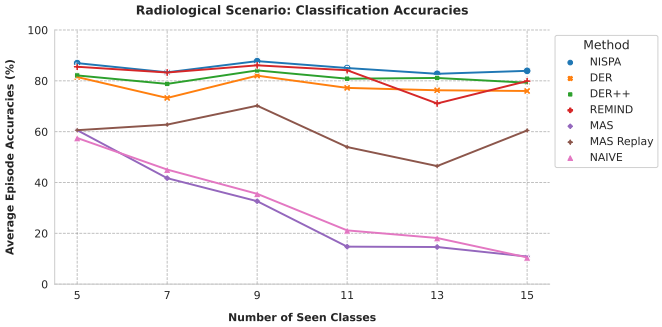

Figure 4: Average accuracy vs tasks seen: Radiological data scenario

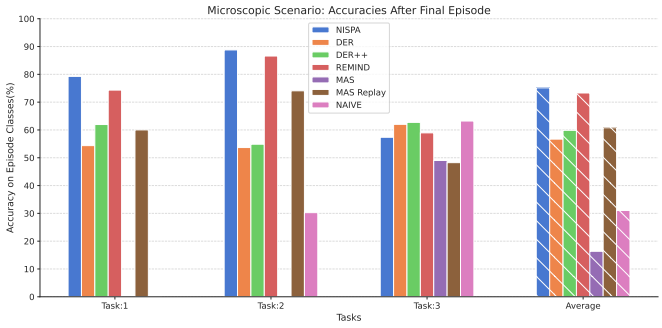

Figure 5: Microscopic data scenario

Student Name	Contributed Aspects	Details
Amritpal Singh	REMIND/Scenario Design/Report	Implementation of REMIND algorithm. Choosing and designing task sequences for scenarios. Help with reporting writing
Mustafa Burak Gurbuz	NISPA/NAIVE/CL settings/Report	Implemented NISPA and NAIVE methods. Created the CL scenarios using the Avalanche framework. Help with reporting and analyzing results.
Shiva Souhith Gantha	MAS/MAS Replay and Report writing	Implemented MAS and MAS Replay methods. Helped with report writing
Prahlad Jasti	DER/DER++ and Report Writing	Implemented DER and DER++ methods. Helped with report writing

Table II: contributions of team members.

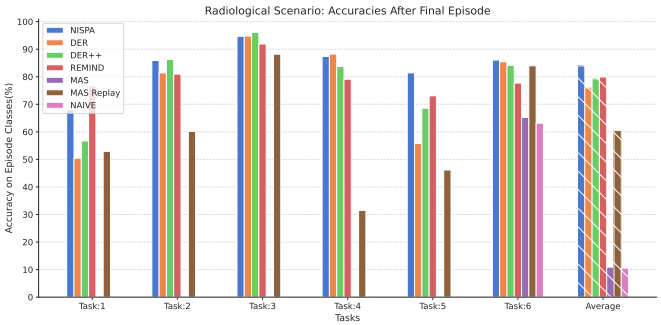

Figure 6: Radiological data scenario

REFERENCES

[1] Andrea Acevedo, Anna Merino, Santiago Alférez, Ángel Molina, Laura Boldú, and José Rodellar. A dataset of microscopic peripheral blood cell images for development of automatic recognition systems. *Data in brief*, 30:105474, jun 2020.

[2] Walid Al-Dhabyani, Mohammed Gomaa, Hussien Khaled, and Aly Fahmy. Dataset of breast ultrasound images. *Data in brief*, 28:104863, feb 2020.

[3] 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)*, September 2018.

[4] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[5] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: A strong, simple baseline. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[6] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[7] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: an extension of MNIST to handwritten letters. *arXiv preprint*, abs/1702.05373, 2017.

[8] Yanming Guo, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu, and Michael S. Lew. Deep learning for visual understanding: A review. *Neurocomputing*, 187, 2016.

[9] Mustafa Burak Gurbuz and Constantine Dovrolis. Nispa: Neuro-inspired stability-plasticity adaptation for continual learning in sparse networks. *arXiv preprint arXiv:2206.09117*, 2022.

[10] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII 16*, pages 466–483. Springer, 2020.

[11] David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[12] Daniel S Kermany, Michael Goldbaum, Wenjia Cai, Carolina C S Valentim, Huiying Liang, Sally L Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, Justin Dong, Made K Prasadha, Jacqueline Pei, Magdalene Y L Ting, Jie Zhu, Christina Li, Sierra Hewett, Jason Dong, Ian Ziyar, Alexander Shi, Runze Zhang, Lianghong Zheng, Rui Hou, William Shi, Xin Fu, Yaou Duan, Viet A N Huu, Cindy Wen, Edward D Zhang, Charlotte L Zhang, Oulan Li, Xiaobo Wang, Michael A Singer, Xiaodong Sun, Jie Xu, Ali Tafreshi, M Anthony Lewis, Huimin Xia, and Kang Zhang. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5):1122–1131.e9, feb 2018.

[13] 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*, 114, 2017.

[14] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521, 2015.

[15] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40, 2017.

[16] Zhiyuan Li, Meiyi Meng, Yifan He, and Yihao Liao. Continual learning with laplace operator based node-importance dynamic architecture neural network. In *International Conference on Neural Information Processing*, 2021.

[17] Vebjorn Ljosa, Katherine L Sokolnicki, and Anne E Carpenter. Annotated high-throughput microscopy image sets for validation. *Nature Methods*, 9(7):637, jun 2012.

[18] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph Lampert. icarl: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[19] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[20] Amir Rosenfeld and John K Tsotsos. Incremental learning through deep adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 42, 2018.

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

[22] Hichem Sahbi and Haoming Zhan. FFNB: forgetting-free neural blocks for deep continual visual learning. *arXiv preprint*, abs/2111.11366, 2021.

[23] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, 2018.

[24] David Silver, Aja Huang, Christopher Maddison, Arthur Guez, Laurent Sifre, George Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529, 2016.

[25] Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning. *arXiv preprint*, abs/1904.07734, 2019.

[26] Xuanang Xu, Fugen Zhou, Bo Liu, Dongshan Fu, and Xiangzhi Bai. Efficient multiple organ localization in CT image using 3D region proposal network. *IEEE transactions on medical imaging*, jan 2019.

[27] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2-a large-scale

lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023.

[28] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations*, 2018.