IDT: an incremental deep tree framework for biological image classification

Wafa Mousser, Salima Ouadfel, Abdelmalik Taleb-Ahmed, Ilham Kitouni

Artificial Intelligence In Medicine 134 (2022) 102392

IDT: An incremental deep tree framework for biological image classification

Wafa Mousser [a,*], Salima Ouadfel [b], Abdelmalik Taleb-Ahmed [c], Ilham Kitouni [d]

[a] Department of Computer Sciences and Applications, Laboratory of Complex Systems' Modeling and Implementation, Abdelhamid Mehri Constantine 2 University, National Biotechnology Research Center Constantine, Algeria

[b] Department of Computer Sciences and Applications, Abdelhamid Mehri Constantine 2 University, Algeria

[c] Institut d'Electronique de Microélectronique et de Nanotechnologie (IEMN), UMR 8520, Université Polytechnique Hauts de France, Université de Lille, CNRS, 59313 Valenciennes, France

[d] LISIA Laboratory "Laboratoire d'Informatique en Science de données et Intelligence Artificielle", "Abdelhamid Mehri Constantine 2 University, Algeria

ABSTRACT

Nowadays, breast and cervical cancers are respectively the first and fourth most common causes of cancer death in females. It is believed that, automated systems based on artificial intelligence would allow the early diagnostic which increases significantly the chances of proper treatment and survival. Although Convolutional Neural Networks (CNNs) have achieved human-level performance in object classification tasks, the regular growing of the amount of medical data and the continuous increase of the number of classes make them difficult to learn new tasks without being re-trained from scratch. Nevertheless, fine tuning and transfer learning in deep models are techniques that lead to the well-known catastrophic forgetting problem. In this paper, an Incremental Deep Tree (IDT) framework for biological image classification is proposed to address the catastrophic forgetting of CNNs allowing them to learn new classes while maintaining acceptable accuracies on the previously learnt ones. To evaluate the performance of our approach, the IDT framework is compared against with three popular in-cremental methods, namely iCaRL, LwF and SupportNet. The experimental results on MNIST dataset achieved 87 % of accuracy and the obtained values on the BreakHis, the LBC and the SIPaKMeD datasets are promising with 92 %, 98 % and 93 % respectively.

Keywords:
Incremental learning
Catastrophic forgetting
Biological image
classification Breast
cancer
Cervical cancer
Convolutional neural
networks

1. Introduction

According to the World Health Organisation, breast cancer is the first most common leading cause of cancer death in women. It is reported that 2.3 million women [1] were diagnosed with breast cancer and 685,000 deaths. In addition, at the end of 2020, there were 7.8 million women alive who were diagnosed with breast cancer in the past 5 years, making it the world's most prevalent cancer. Likewise, cervical cancer is a real health problem; it is the fourth cancer causing death in females with an estimated 604,000 new cases and 342,000 deaths in 2020 [2].

Unlike the cervix cancer which is caused by infection with human papillomaviruses (HPV), the breast cancer is due to other factors such as patient's age, family history and previous benign breast lump. Even if effective primary HPV vaccination prevents most cervical cancer cases, the screening and treating precancerous lesions remain the best prevention approaches for both breast and cervical cancers. According to the International Agency for Research on Cancer (IARC-2020), breast cancer has overtaken lung cancer as the world's most commonly diagnosed cancer. Early diagnosis significantly increases the chances of correct treatment and survival, but this process is tedious and often leads to a disagreement among pathologists [3]. Computer-Aided-Diagnosis (CAD) systems showed potential improvement of the diagnosis accuracy.

Data mining and machine learning techniques are transforming the decision-making process in the medical world [4] and extensive research efforts have been carried out in the area of accurate and early diagnosis of cancers. The reinforcement learning, as a branch of machine learning, has been used with supervised learning in an adaptive online learning framework to support the breast cancer clinical diagnosis [5]. Likewise, the reinforcement learning has been used to identify breast lesions from magnetic resonance images [6], and in the automatic inverse treatment planning for cervical cancer High Dose-Rate brachytherapy [7]. In addition, the reinforcement learning allows the localisation of any organ in a Computed Tomography scan [8] and shows thus, an accurate breast

* Corresponding author.
E-mail addresses: wafa.mousser@univ-constantine2.dz (W. Mousser), salima.ouadfel@univ-constantine2.dz (S. Ouadfel), taleb@uphf.fr (A. Taleb-Ahmed), ilham.kitouni@univ-constantine2.dz (I. Kitouni).

cancer risk assessment [9]. Further, the histopathology classification and the microarray technology have been proven to be effective in breast cancer [10].

Despite the amount of study which use traditional machine learning techniques for predicting cancer survival, researchers are now moving towards deep learning and hybrid approaches to gain some insights into survival prediction [4].

In recent years, deep learning methods in general and Convolutional Neural Networks (CNNs) in particular have been used in many fields: pattern recognition, computer vision, natural language processing and speech recognition [11]. Due to the exponential increase of data, CNNs produce results similar, if not better than human expert performance not only in the above domains but also in bioinformatics, drug design and others [12].

In medical image analysis, deep learning has been used for the automated nuclei segmentation and classification in cervical cells from pap smear images [13]. Likewise, the CNNs allow the intelligent diagnosis of cervical pre-cancerous lesions [14] using automated feature extraction. More recently, combining the colposcopy based deep learning model, the cytology test results, and the HPV test results, improves significantly the cervical screening accuracy [15]. Further, the performance of deep learning algorithms is evaluated to detect breast cancers on chest CT [16]. In addition, the application of deep learning in breast cancer imaging has been overviewed in [17] where, it has been stated that histopathological imaging is the most common approach used to detect breast cancer.

However, the success of CNNs in computer vision applied on bioinformatics especially for CAD relies on the fact that they operate on entire datasets at once without considering the situation where the dataset dynamically grows over time [11].

Currently, most biological datasets are difficult to access due to the patient's data privacy on one hand. On the other, the amount of these datasets is regularly growing and the number of classes is continuously increasing. Retraining a CNN from scratch using old and new data is highly computationally costly. Building powerful models is no longer a difficult task, but updating their knowledge is a challenging one [18]. Combining deep learning techniques with incremental learning methods allows the resulting models to learn continuously, accumulate the knowledge learned in previous tasks, while using it to help future learning.

Incremental Learning, sometimes also called Continual Learning or Lifelong Learning, is considered as a training procedure where the objective is to train a model, previously built to perform a set of tasks, to learn new tasks and allow it to get high performances on both new and old tasks [19]. However, incremental algorithms are prone to forget previously learned knowledge when adapted to a new task, known as *catastrophic interference* or *forgetting* [20]. Besides this issue, many challenges face an incremental learning algorithm such as online model parameter adaptation caused by the fact that datasets are not available initially, but arrive over time as for biomedical data. Another challenge is known as *the stability-plasticity dilemma* where learning new tasks requires plasticity for the integration of new knowledge, but also stability in order to prevent the forgetting of previous knowledge. *Concept drift* and *model benchmarking* remain also challenging issues for incremental learning.

Continual learning covers three scenarios: *Task Incremental Learning*, *Domain Incremental Learning*, and *Class Incremental Learning*. The difference consists mainly on whether at test time, information about the task identity is available, and if not, whether the model is also required to explicitly identify the task to solve [21]. The *Task-IL* is considered as the easiest scenario for incremental learning and relies on a typical network architecture shared for all the tasks and a target **multi-headed** output layer for each one of them. This scenario is used when the task identity is provided and models are informed about which task needs to be performed. The *Domain-IL* scenario is considered when the task's structure is always the same but the input distribution is changing, while the task identity is not available at test time. In such a situation, the models are not required to infer which task it is and, need however, to solve the task at hand. The *Class-IL* scenario is considered in the situation where models have to incrementally learn new classes of objects. They must be able to both solve each task seen so far and infer which task they are presented with. Even if the number of new strategies is increasing in each category, few techniques are explored at the intersection of the three categories.

Based on how task specific information is stored and used throughout the sequential learning process, the proposed strategies to tackle the catastrophic forgetting while learning incrementally a CNN are divided into three families [19,21,22] namely the *Replay-based*, *Regularization-based* and *Parameter isolation-based* methods. In the first, catastrophic forgetting is mitigated by relying on storing previous experience explicitly or implicitly. The *Regularization-based* methods are used when no storage of raw input is possible, mainly motivated by privacy reasons, as in the case of medical applications. These methods are memory efficient and propose, in general, an extra regularization term in the loss function to consolidate previous knowledge when learning new data. Last, the *Parameter isolation-based* methods are used when optimal performance is a priority and the model's capacity is not constrained on the architecture's size. These methods are better suited for learning a long sequence of tasks and perform by freezing the set of parameters learned after each task and growing new branches dedicated for new tasks.

Despite the availability of such continual learning techniques in computer vision in general, applying them for biological data sets does not provide such precise results, particularly with the increase of the number of classes, where updating the models requires the presence of both new and old datasets. Furthermore, the limited amount of labelled data for training presents problems in biomedical domains. Paradoxically, this amount of data is regularly growing and the number of classes is continuously increasing. Thus, training deep models from scratch based on both old and new data is highly computationally costly, in addition to legal and privacy constraints on work with sensitive health records [23].

In this paper, we present an *Incremental Deep Tree framework for biological image classification* named the IDT framework. This framework is built in a tree-like manner, where in order to learn a new class, a new model is constructed by adding a branch linked to each previously learned task. These branches are used to regularise previously acquired capabilities based on a set of replay-data to alleviate and overcome catastrophic forgetting. Thus, knowledge is updated, and the root model can predict new classes without forgetting old ones by maintaining acceptable accuracies for each one of them.

The main contributions of the IDT are the fact that firstly the model does not need all previously learned data. Hence, it is able to update its knowledge without being trained from scratch. Another novelty regards the hierarchical classification which guides the model to predict image labels based on the selected branch of the tree. Finally, the proposed approach combines the three continual learning strategies as shown in Fig. 1, where some of the most popular continual learning strategies are presented. The IDT is situated in the intersection of the red, the green and the blue circles for the Replay, the Regularization and the Parameter-isolation based methods respectively.

In the second section of the paper, we present a review of related works to the incremental learning of CNNs, with particular focus on bioinformatics. We devoted the next sections to the problem formulation, the description of different components of the proposed framework, the experimented datasets, the compared methods and metrics, and a discussion of the obtained results respectively. Finally, we conclude with the significance of this work and suggest the promising perspectives in this domain.

2. Related works

Replay-based methods: *Rehearsal methods* are referred to when a

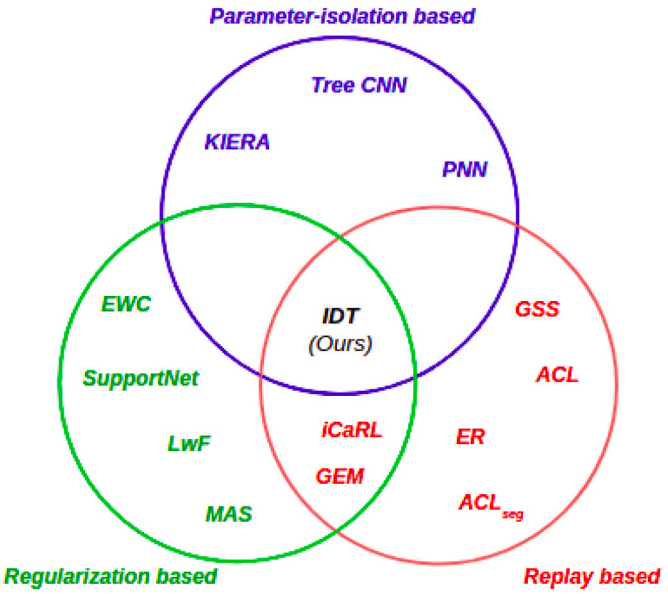

Fig. 1. Venn diagram of the most popular continual learning strategies. Replay-based methods (red): GSS [30], ACL [28], ER [25], ACLSeg [27], Parameter isolation-based methods (blue): KIERA [40], Tree-CNN [38], PNN [39], Regularization-based methods (green): EWC [33], SupportNet [18], LwF [31], MAS [35].
iCaRL [24] and GEM [29] relies on both Replay based and Regularization-based methods.
IDT(ours): combines the three continual learning strategies. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

subset of stored data is explicitly used to retrain the model while training on a new task such as iCaRL [24] and ER [25]. The former replays stored data as well as the current task input with a special form of distillation while training a feature extractor. The latter suggests the use of reservoir sampling to limit the number of stored samples to a fixed budget assuming an overall i.i.d. distributed data stream. *Rehearsal methods* are constrained to a fixed budget to accommodate new classes, hence might be prone to overfitting the subset of the stored samples. When previous samples are no longer available, *pseudo-rehearsal methods* could be used to generate high-quality samples as proposed in DGR [26], ACLSeg [27] and ACL [28], where in the latter, raw samples are used by the model's parts (discriminator and shared-module) to prevent forgetting in the features found to be shared across tasks. The major drawback of these methods is the complexity of training the generative model continually. In fact, the methods that *constraint the optimisation* rely on backward/forward transfer and the key idea is to only constrain the update of the new task not to interfere with the previous ones as proposed in GEM [29]. Moreover, selecting a subset of samples that maximally approximate the feasible region of the historical data such as in GSS [30] leads to a pure online continual learning setting where no task boundaries are provided. In this work, we explicitly use a *random-data-selector* to store pre-processed samples for the replay, which constraint the updated model to recognise old classes when training on new ones.

Regularization-based methods: are used when the replay of stored samples from previous tasks is not possible. The current family encompasses *data-focused* and *prior-focused methods*. In the former, knowledge distillation from the old model to the new one is the base block such as in LwF [31] and DMC [32] where the output of the old model is used as soft labels for previous tasks to mitigate forgetting. In SupportNet [18], regularization parameters and support vectors are used to mitigate forgetting during the class incremental learning. These methods are prone to the concept drift when the domain shifts between tasks. However, when learning new data, the *prior-focused methods* relies on an estimated distribution over the model parameters to mitigate forgetting [19]. This is obtained by penalising the change of important parameters when training the model on new tasks such as in EWC [33]. As an extension to EWC, Laplace approximation is put forward to construct the parameter importance matrix in [34], and the use of unsupervised and online criterion for the modification of parameter importance matrix is proposed in MAS [35]. Recently, an inter-task synaptic mapping is proposed in [36]. Even if the *regularization-based* methods are computationally efficient, these methods require the task boundaries and task IDs to be known. In this work we set up a branch for each previously learned task while learning new classes. This branch allows the updated framework to recognise images from old classes.

Parameter isolation-based methods: are used when learning a long sequence of tasks in the *Task-IL* scenario, especially when the model's capacity is not constrained and optimal performance is a priority. While training the model on a new task, the *fixed-network methods* mask the model's parts dedicated for previous tasks and mainly use a task oracle for predictions such as the learn-to-grow framework [37] where the neural architecture search is used to find a network structure that better handles each task. However, the *dynamic-architecture methods* are used when the size of the model's architecture is not constrained and proceed by freezing the set of parameters learned after each previous task then grow new branches for new tasks, such as in Tree-CNN [38] which proposes a self-growth in a tree-like manner of the network based on features hierarchy reorganisation when new tasks arrive. The PNN [39] introduces a new column for every new task while freezing old network parameters. KIERA [40] uses a flexible deep clustering approach possessing an elastic network structure to cope with changing environments in the timely manner. To overcome catastrophic forgetting, the centroid based experience replay is put forward. The main limitation of the *parameter isolation-based methods* are their high computational and memory burdens. In this work, we implicitly use the *parameter-isolation* by fixing the architecture and the parameters of the old task's networks, coupled with a tree-like *dynamic-architecture* to generate and learn the updated framework.

3. Problem formulation

In this paper, we propose a framework that learns incrementally deep models for biological image classification. In our setup, the training data $\mathscr{D}$ is organised into $\mathscr{B}$ batches, with $\mathscr{D} = \{\mathscr{B}_t\}_{nbr\_batchs}^{1}$, where the number of batches is continuously growing.

Each batch $\mathscr{B}_t$ consists of $\mathscr{I}_n$ labelled training data i.e., $\mathscr{B}_t = \left\{\left(x_j, y_j\right)\right\}_{j=1}^{N_t}$, where $x_j \in \mathbb{R}^d$, y_j is a discrete label and $\mathscr{N}_t$ is variable across sessions. This framework learns sessions sequentially starting by the first batch in the *Learn mode* (*sec* 4.2.1.1). However, the model can be evaluated on test data among sessions using the *Predict mode* (Section 4.2.1). The class incremental learning scenario presented in Fig. 2 shows that at T_0, on the left, the *Base_Model* learns the available dataset $Batch_0$ as described in the paragraph *"Learning the first batch"* (Section [par: Learning the first batch]). Sequentially, the framework generates a new model $Model\_Batch_i$ for each new batch of data, using $\mathscr{I}_n$ classes and additional branches to replay each previously learned task as detailed in the paragraph *"Learning next batches"* (Section [par: Learning next batches]).

4. Method

Human brain seems to have the remarkable ability to learn a large number of different tasks without getting any of them negatively interfering with the others [41]. A person needs only a part of previously acquired knowledge to learn new skills. These information (knowledge) guide it to search and check differences among old and new tasks. The common strategy for the human brain to overcome forgetting during

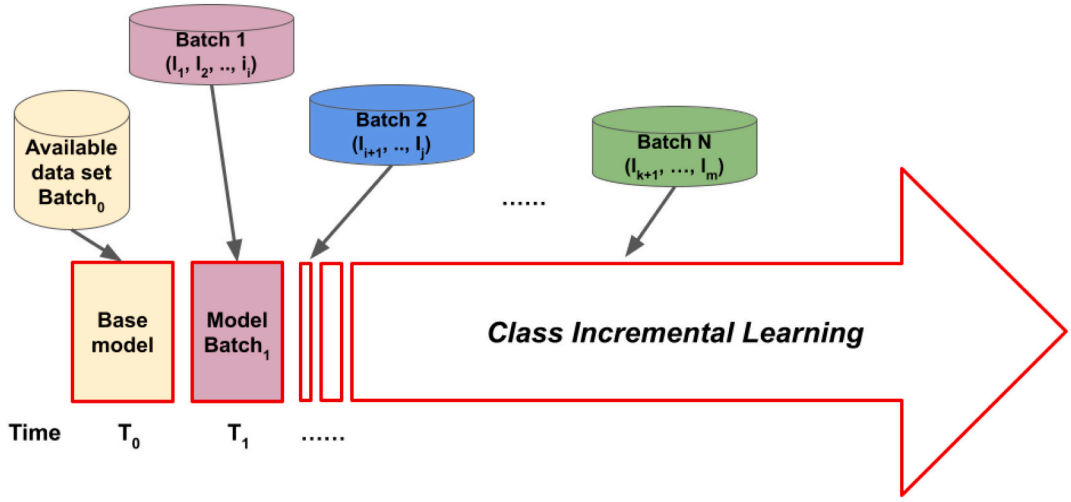

Fig. 2. Class incremental learning scenario. When a dataset $Batch_i$ is available, a new model Model $Batch_i$ is trained to learn the classes belonging to $Batch_0$, …, $Batch_i$.

learning is to review the old knowledge frequently [18]. In practice, human usually does not learn from scratch and does not review all the details, but rather the important ones, which are often sufficient for him to grasp the knowledge [18]. Consequently, humans never learn in isolation. In contrast, they always retain the knowledge learned in the past and use it to help future learning and problem solving. For this reason, detecting the most important differences between old and new tasks helps considerably the learner and reduces the learning cost.

Based on the above findings, the key idea of our method is biologically inspired. The IDT framework relies on two modules namely the *Pre-processing* and the *Incremental tree (IT)* as shown in the Fig. 3. The former is responsible for image pre-processing including augmentations, feature extraction and data selection that allows the models to review the old knowledge while learning new classes. Whereas, the latter (*IT*) is devoted to generate and train new models on one hand, and growing the architecture in a tree-like manner on the other hand, allowing the models to differentiate old and new classes.

4.1. Pre-processing module

4.1.1. Augmentations

Despite the fact the deep models have achieved state-of-the-art results in many computer vision tasks, CNNs are still challenged by the limited size of the available databases at the learning phase, hindering the potential for generalisation of knowledge. Such a situation is known as overfitting. To overcome this challenge, we bring the microscopy images into a common space and enable improved quantitative analysis, using the unsupervised feature representation as proposed in [3] to obtain normalized images.

The normalisation process includes the following sequence: a random colour augmentation is performed for each image, then the magnitude of every pixel is multiplied by two random uniform variables from the range(0.3, 1.7) see Fig. 17(b). Finally, the set of the obtained descriptors is combined through p-norm pooling into a single descriptor as shown in the *Eq. 1*, where d_{pool} is the pooled descriptor of the image and N is the number of crops. It could be noted that d_i is the descriptor of the i^{th} crop and p is set to 3 as suggested in [3]. The obtained images from the Augmentation submodule are used in the next phase.

$$d_{pool} = \left(\frac{1}{N} * \sum_{i=1}^{N} (d_i)^p \right)^{\frac{1}{p}} \tag{1}$$

4.1.2. Feature extraction

The feature extraction is the second step of the pre-processing module. Commonly defined as the process of dimensionality reduction by which the initial set of raw data (image pixels) is reduced to more manageable groups for processing. For this purpose, we use the standard

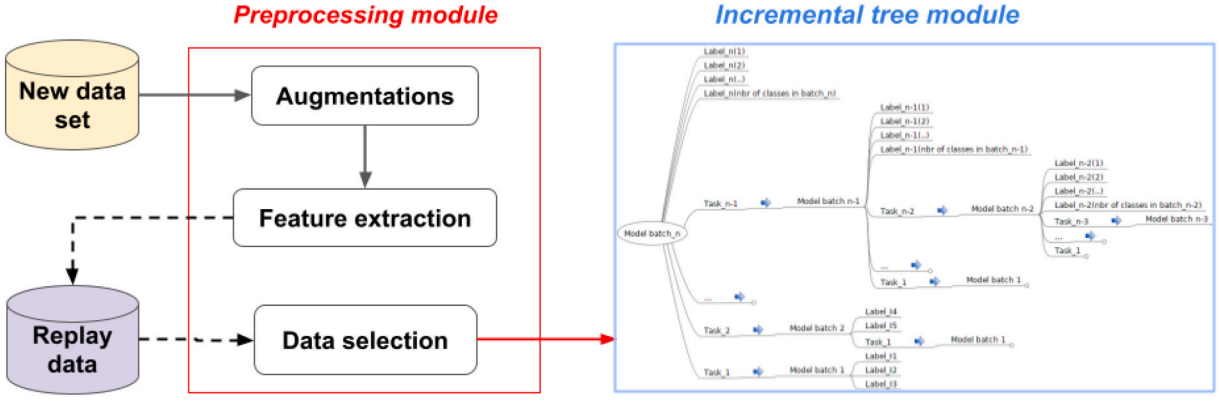

Fig. 3. The architecture of the proposed framework. It consists of a **pre-processing module** (red) which is used for data augmentation, feature extraction and data selection the **incremental tree module** (blue) which is an extensible fully connected layer classifier with an output node for each previously learnt Task. The **Data selection** provides a part of old data to train the new *Model Batch on old Tasks*.

pre-trained ResNet50 from Keras distribution [42] where fully connected layers are removed from the model as presented in [43] and channels of the last convolutional layer are converted into a one-dimensional feature vector of 2048 size as shown in Fig. 4.

4.1.3. Data selection

An incremental deep model requires plasticity for the new knowledge integration and stability to prevent knowledge forgetting [44]. In the field of continual learning this is a well-known challenge faced by deep models *"the stability-plasticity dilemma"*. In fact, too much plasticity leads to a constant gradual loss of previously acquired knowledge *"the catastrophic forgetting"*, whereas too much stability will bias or bound learning new abilities.

To address the above challenges, various methods have been proposed such as the iCaRL [24], the ACL [28] and the SupportNet [18], where the focus was mainly the representative quality of the selected images, which is out of the scope of this paper.

In the IDT framework *data selection*, the old images used when training new models on next batches were randomly chosen by the use of a random *data-selector* which manages a replay-data memory from each previously learnt class. Given that the total memory size is not expandable, the required number of images from old classes should decrease when new batches are added. The *data-selector* in the IDT framework uses the Formula Fig. 5 to compute the required size of memory to be allocated to each class when learning the current batch.

$$nbr\_images\_old\_classes = \frac{mean\left(nbr\_images\_current\_Batch\right)}{nbr\_classes\_in\_old\_Batch} \tag{2}$$

where

$$mean\left(nbr\_images\_current\_Batch\right) = \frac{nbr\_images\_current\_Batch}{nbr\_classes\_current\_Batch} \tag{3}$$

Except the first iteration, the normal process for pre-processing module awaits the completion of both augmentation and feature extraction before running the *data selection* module. At iteration 01, the new dataset is considered as a *replay-data*. In the next iterations, the *data-selector* reduces the memory size of old classes as described by the equation above.

4.2. Incremental tree module

The main contribution of this paper is the generation of a predictive model in a tree-like manner. For each learning session (round), a new fully connected classifier is built with outputs related to the class labels available in the new batch. However, for each old batch, a linked label (*Task_i*) is included in the output layer as presented in Fig. 5. Then, the additional branches, related to *Tasks*, are trained using the images obtained from the *data selection* module. Actually, the *Model_Batch_n* outputs are all the classes available in the *batch_n* and also *Task* labels related to each old batch, allowing the predictive model to prevent catastrophic forgetting.

Further, since the CNNs architecture and hyper-parameters is out of the scope of the proposed method, the IDT acts only on the output layers of deep classifiers and does not focus on their inner architecture. This gives flexibility and modularity in terms of the CNN architecture used for different *Tasks* and opens avenues for the IDT application in other fields.

In practice, the IDT can operate in two different modes namely the *Learn mode* and the *Predict mode*. In the former, the framework is used to update knowledge and to extend the classes that the root model is able to recognise. In the latter, the IDT relies on the models for the class label prediction of test images. Consequently, depending on the *mode* variable content, the framework is used as presented in Algorithm 1.

Algorithm 1: Main tree

Input: $\mathcal{D} = [\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_{nbr\_batchs}]$;
$mode$;
Begin
if $mode == Learn$ **then**
  $Learn\_mode()$;
else if $mode == Predict$ **then**
  $Y\_pred \leftarrow Predict\_mode()$;
end
End

The structure of the incremental tree module. *Model_batch_n* is a fully connected classifier. *Label_i* is a class name. A branch corresponds to an output neuron and the blue arrows refer to old classifiers.

4.2.1. Learn mode

When the algorithm is used in *Learn mode*, the last configuration of the tree and the replay-data are loaded. At the end of the first iteration, in Algorithm 2, the root of the tree is a fully connected classifier which recognizes images belonging to *Batch_1*. However, if the tree is not empty, a new classifier for the current batch of images is built and fitted

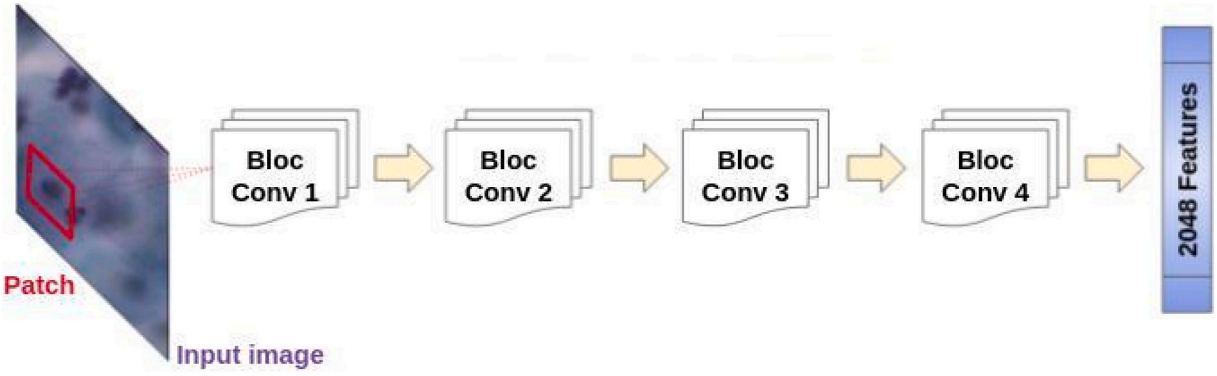

Fig. 4. Feature extraction using the pre-trained model ResNet50.

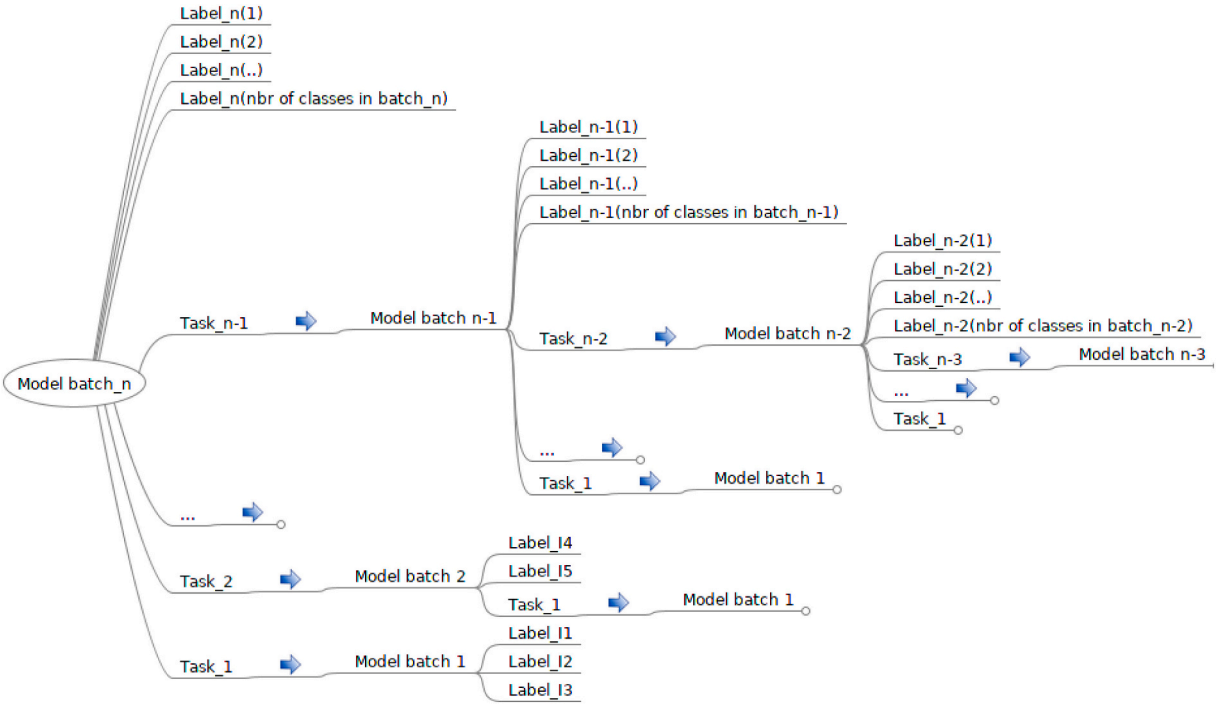

Fig. 5. The structure of the incremental tree module. Model batch_n is a fully connected classifier. Label_i is a class name. A branch corresponds to an output neuron and the blue arrows refer to old classifiers.

using pre-processed data (Section 4.1). This model has in the last fully connected layer neurons related to previously learnt batches, allowing it to distinguish images belonging to the current batch from those of old batches.

Moreover, if the number of available batches is greater than one, then the *Learn mode*, loops over these batches and uses pre-processed data (Section 4.1) jointly with replay-data to train the new classifier. Finally, the tree structure is updated, and the IDT can accordingly be used in either *Learn* or *Predict* mode. The architecture of the fully connected layer model is presented in Section 5.3.

(*Batch*_1) ≥ 2. Therefore, the tree structure is generated when a dataset of at least one batch is available as presented in the Figure 5 where, a *Batch* refers to a set of labelled images belonging to distinct classes, and the *Model_batch_n* is a fully connected classifier trained on images belonging to the *Batch_n* and the replayed data from *Batch*_($n-1$), . . , *Batch*_(1). In this way, the pre-processing module detailed in the Section 4.1 is performed on *Batch*_1 = [*Label*_1, *Label*_2, *Label*_3].

Next, the *Learn mode* builds and trains from scratch the deep classifier *Model_batch*_1 with 3 neurons in the last fully connected layer. After convergence, *Model_batch*_1 is considered as the root of the incremental tree and can accordingly be used in the *Predict mode* (Section 4.2.1.1) where for each test image, the *Model_batch*_1 returns a class label belonging to *Label*_1, *Label*_2 or *Label*_3.

Algorithm 2: Learn mode

Input: $\mathcal{D}$;
Begin
$Tree \leftarrow Load\_tree()$;
$Old\_batches \leftarrow Load\_Old\_batches$;
if *Tree is empty* **then**
  $\mathcal{B} \leftarrow first\_batch(\mathcal{D})$;
  $\mathcal{X}\_train, \mathcal{Y}\_train \leftarrow Preprocess(\mathcal{B})$;
  $\mathcal{M} \leftarrow Base\_Model(nbr\_outputs = len(\mathcal{B}))$;
  $\mathcal{M}.train(\mathcal{X}\_train, \mathcal{Y}\_train)$;
  $Save\_model(\mathcal{M})$;
  $\mathcal{D}.pop(\mathcal{B})$;
  $Tree.Update()$;
end
for $\mathcal{B} \in \mathcal{D}$ **do**
  $\mathcal{X}\_train, \mathcal{Y}\_train \leftarrow Preprocess(\mathcal{B})$;
  $\mathcal{M} \leftarrow Base\_Model(nbr\_outputs = len(\mathcal{B}) + len(Old\_batchs))$;
  $\mathcal{M}.train(\mathcal{X}\_train, \mathcal{Y}\_train)$;
  $Save\_model(\mathcal{M})$;
  $Tree.Update()$;
end
End

4.2.1.1. Learning the first batch. The IDT starts learning the first batch of data *Batch*_1 which must contain at least two classes, i.e.: *len*

4.2.1.2. Learning next batches. In the next iteration, when the *Batch*_2 = [*Label*_4, *Label*_5] is added, as shown in the Fig. 1, the *Learn*

mode builds the *Model_batch_2* which is trained from scratch using the *Batch_2* preprocessed data (Section 4.1) jointly with the *Batch_1* replay-data.

Consequently, the IDT replays *m* images from each class previously learned i,e.: *Label_1*, *Label_2* and *Label_3*, where $m = mean(nbr\_images\_batch\_2)/3$, (see Eq. 3). Thus, the root of the incremental tree is changed to *Model_batch_2* and the framework can perform predictions (*Predict mode*).

When the $Batch\_3 = [Label\_6, Label\_7, Label\_8]$ is added as in Fig. 5, the *Learn mode* builds and learns from scratch the *Model_batch_3* using the *Batch_3* preprocessed data, jointly with a replay-data from both *Batch_1* and *Batch_2*.

The incremental tree root is then set to *Model_batch_3*. In the latter, the number of neurons in the output layer is equal to 5, where the 3 first ones correspond to *Label_6*, *Label_7* and *Label_8*. However, the 2 remaining neurons refer to the replayed images from the *Batch_1* and the *Batch_2* labelled respectively as *Task_1* and *Task_2* in the Fig. 5.

4.2.2. Predict mode

As mentioned above, the IDT can switch into the Predict mode after each Learn mode (section [subsection: Learn mode]). For this purpose, the labels of *X_test* images are predicted using the *Model_batch* present in the root of the tree as presented in the Algorithm 3.

Referring to the example of Fig. 1, for each test image, the IDT checks whether the obtained label is *Label_6*, *Label_7* or *Label_8*. If not, the IDT relies on the *Model_batch* related to the output *Task* to predict the corresponding class name for test images, namely *Model_Batch_1* for *Task_1* and *Model_Batch_2* for *Task_2*. The blue arrows in Fig. 1 indicate that if the *Model_Batch_n* returns *Task_i*, then the IDT framework uses the *Model_Batch_i* to predict the corresponding label.

In contrast with federating learning which aims to combine learned knowledge from non-co-located data, the addressed datasets are centralised and the current predictive model is not averaged using learner's updates. In addition, in the proposed framework, the old models are static predictive models and their knowledge does not improve when new batches of data are available. Therefore, the old models are exclusively utilised in the *Predict mode*, and the framework updates its knowledge at the root-level as presented in the Paragraph [par: Learning next batches].

Algorithm 3: Predict mode

Input: $\mathcal{D}, X\_test$
Result: Y_pred
Begin
$Model \leftarrow Root\_Model$;
$Y_{pred} \leftarrow Model$.predict($X\_test$);
for $i \in range(len(X\_test))$ **do**
  if $Y\_pred[i] = Task\_label$ **then**
    $Y\_pred[i] \leftarrow Model\_Task\_label$.predict($X\_test[i]$);
  end
end
return Y_pred;
End

5. Experimental results

5.1. Data

5.1.1. MNIST dataset

The *Modified National Institute of Standards and Technology* database of handwritten digits known as MNIST [45] is a large database commonly used for learning techniques and pattern recognition methods on real-world data while spending minimal efforts on pre-processing and formatting. It is also used for training various image processing systems. The MNIST database has a training set of 60,000 examples and a test set of 10,000 examples. It is a subset of a larger set

Fig. 6. Samples from the MNIST dataset.

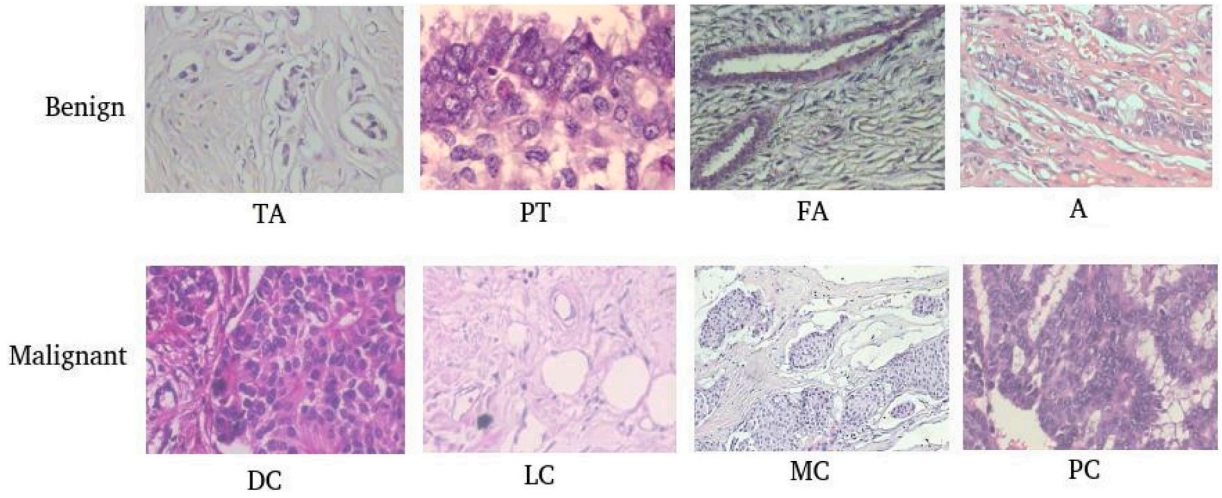

Fig. 7. Samples from the BreakHis dataset. The magnification factor is 400×. The **benign** breast tumours include: tubular adenoma (TA), phyllodes tumour (PT), fibroadenoma (F), and adenosis (A). The **malignant** tumours or breast cancer include: ductal carcinoma (DC), lobular carcinoma (LC), mucinous carcinoma (MC) and papillary carcinoma (PC).

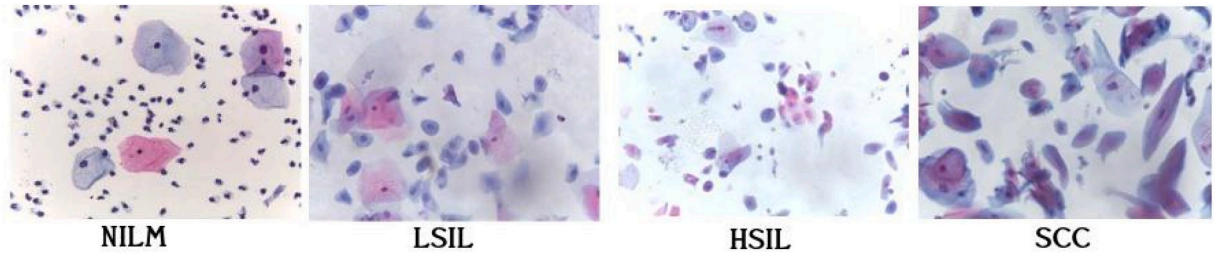

Fig. 8. Samples from the LBC Pap smear dataset. **Normal** category includes the Negative for Intra-Epithelial Malignancy (NILM). **Abnormal** category includes: Low Squamous Intraepithelial Lesion (LSIL), High Squamous Intra-Epithelial Lesion (HSIL) and Squamous Cell Carcinoma (SCC).

Fig. 9. Samples from the SIPaKMeD dataset.

available from NIST [46]. Fig. 6 shows samples from the MNIST dataset, where digits have been size-normalized in a fixed-size image of 28 × 28 pixels.

5.1.2. BreakHis dataset

The Breast Cancer Histopathological Image Classification (BreakHis) database has been built in collaboration with the P&D Laboratory Pathological Anatomy and Cytopathology, Paraná, Brazil. It is composed of 9109 microscopic images of breast tumour tissue collected from 82 patients using different magnifying factors 40×, 100×, 200×, and 400×. It contains 2480 benign and 5429 malignant samples (700 × 460 pixels, 3-channel RGB, 8-bit depth in each channel, PNG format).

The dataset is divided into two main groups: benign tumours and malignant tumours. Histologically benign cells refer to lesions that does not match any criteria of malignancy, while malignant refers to cancer cells. The dataset contains four histological distinct types of benign breast tumours adenosis (A), fibroadenoma (F), phyllodes tumour (PT), and tubular adenoma (TA); and four malignant tumours (breast cancer): ductal carcinoma (DC), lobular carcinoma (LC), mucinous carcinoma (MC) and papillary carcinoma (PC) as shown in Fig. 7.

5.1.3. Pap smear datasets

Liquid based cytology is one of the cervical screening tests. The repository consists of a total of 963 images sub-divided into four sets of images representing the four classes of pre-cancerous and cancerous lesions of cervical cancer as per standards under The Bethesda System [47]. The NILM or normal category contains 613 images, while the 350 remaining images belong to the abnormal category (see Fig. 8). The Pap Smear images were captured in 40× magnification from 460 patients then collected and prepared using the liquid-based cytology technique.

The SIPaKMeD Database consists of 4049 images of isolated cells that have been manually cropped from 966 cluster cell images of Pap smear slides [48]. These images were acquired through a CCD camera adapted to an optical microscope. The cell images are divided into five categories containing Superficial-Intermediate cells, Parabasal cells, Koilocytotic cells, Dysketarotic cells, and Metaplastic cells, see Fig. 9.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 26, 26, 32)	320
conv2d_1 (Conv2D)	(None, 24, 24, 64)	18496
max_pooling2d (MaxPooling2D)	(None, 12, 12, 64)	0
dropout (Dropout)	(None, 12, 12, 64)	0
flatten (Flatten)	(None, 9216)	0
dense (Dense)	(None, 128)	1179776
dropout_1 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 2)	258

Total params: 1,198,850
Trainable params: 1,198,850
Non-trainable params: 0

Fig. 10. The CNN architecture for MNIST dataset.

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 1024)	2098176
dropout (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 512)	524800
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 2)	1026

Total params: 2,624,002
Trainable params: 2,624,002
Non-trainable params: 0

Fig. 11. The CNN architecture for biological datasets.

5.2. Compared methods and metrics

Experiments of our proposed method have been carried out at the *High Performance Computing Constantine* using a 32 IBM X3550 M4 nodes, with 02 Intel Xeon 08 cores processors and 24 GB RAM for each one of them.

First, our framework is compared against the iCaRL, LwF, the *fine-tuning* and the *all data configuration* on the MNIST database. The *all data configuration* refers to a training from scratch using all the classes of new and old batches as a single dataset. The number of neurons in the output layer of the model is equal to the number of classes in the entire dataset, which allows the training of a simple fully connected classifier. Such a configuration is used as an upper-bound experimentation. In contrast, the *fine-tuning* refers to a lower bound experimentation in which old models are re-trained using the current batch of data exclusively.

For the BreakHis dataset the proposed framework is compared to iCaRL, SupportNet and the *all data configuration*. To the best of our knowledge, the incremental learning of deep models has not been addressed in the field of cervix cancer. For this reason, the Pap Smear and the SIPaKMeD datasets are compared against the *all data configuration*.

After each experimentation session (round), we focus on the accuracy of the learned classes. The figures (Fig. 15, Fig. 16, Fig. 20 and Fig. 21) are thus obtained. The total number of classes and the achieved accuracies are represented on the X and Y axes respectively. The average accuracy for each experimentation is reported between brackets and the *Cumulative model* refers to the *all data configuration*.

The next metric of interest is the class precision achieved by the IDT,

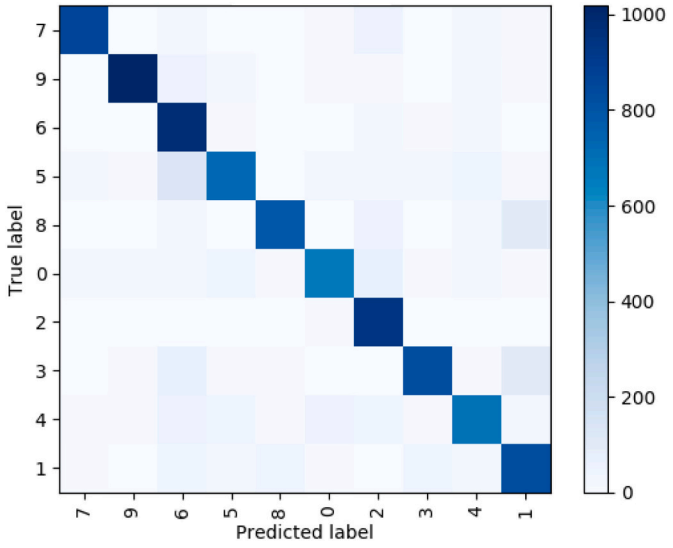

Fig. 12. The IDT confusion matrix on the MNIST dataset.

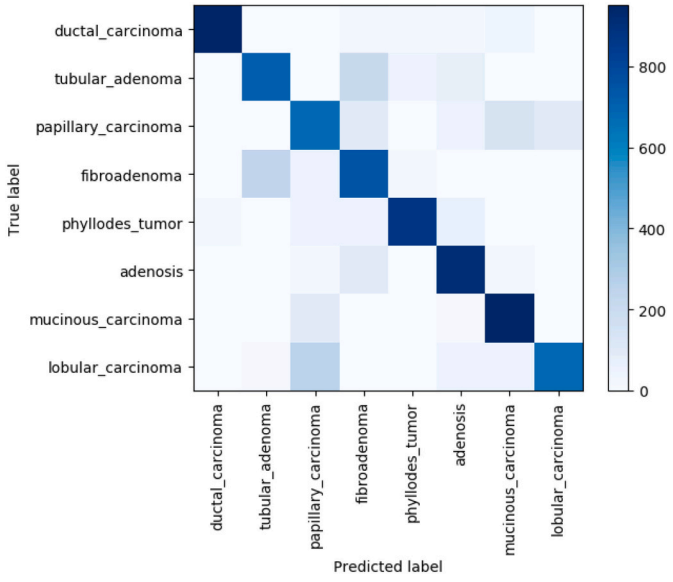

Fig. 13. The IDT confusion matrix on the BreakHis dataset.

where the figures (Fig. 14, Fig. 18 and Fig. 19) show the class names and the corresponding precisions on the X and the Y axes respectively. In the above figures, the blue cumulative model bars refer to the *all data configuration* experimentations. Furthermore, the confusion matrices of the IDT predictions are presented in figures (Fig. 12, Fig. 13, Fig. 22 and Fig. 23).

As mentioned in the Section 4.2, since the IDT framework does not rely on a specific architecture of CNNs, a fixed base model is used while experimenting with a database.

For the biological databases, the sequential model proposed in [43] is used to build a Multi-Layer Perceptron (MLP). The grid search study, performed in the above work, fixes the hyper-parameters as presented in Fig. 11 which includes three fully connected layers. The input layer contains 1024 neurons with $input\_size = 2048$, followed by 512 neurons with a *ReLU activation* and *0.5 Dropout* for each one of them. Finally, the model outputs N neurons with *softmax activation*, where N refers to the number of classes that the model attempts to learn.

For the MNIST database, authors used a simple convnet that achieved almost %99 test accuracy as proposed in [49] (See Fig. 10). Thus, when a new classifier is built, the IDT sets the number of neurons in the last fully connected layer as presented in the Algorithm2.

All models in the tree are separately trained for 100 epochs with 32 images per batch using the *Adam* optimiser and the *categorical cross-entropy loss* function. In addition, the *early stopping* function is used to abort the training process if the validation accuracy stops improving with a patience fixed to 10.

5.3. Obtained results and discussion

5.3.1. MNIST dataset

Unlike biological databases, the MNIST dataset images are normalized and centred. In fact, if the IDT framework augmentations were applied to such images, the obtained features would not contain relevant information. For this reason, the proposed pre-processing module is ignored. For the MNIST dataset, the 10 classes are fed in 5 batches and each one of them contains 2 classes. The experiments show that neither the class order nor the batch order affects the final precision. Moreover, in contrast with the incremental learning methods for which the prediction precision decreases after each learning session (round), the precision achieved by the IDT prediction increases after learning new classes.

Fig. 14 shows a performance comparison on the MNIST dataset and the following points were observed. The precision achieved by the $Model\_Batch_4$ on the class "0" reaches 95 %, while the $Model\_Batch_3$ performed almost 85 % for the same class.

The same findings are drawn for other classes such as: "1", "3" and "7", where the precisions achieved by the $Model\_Batch_3$, $Model\_Batch_4$

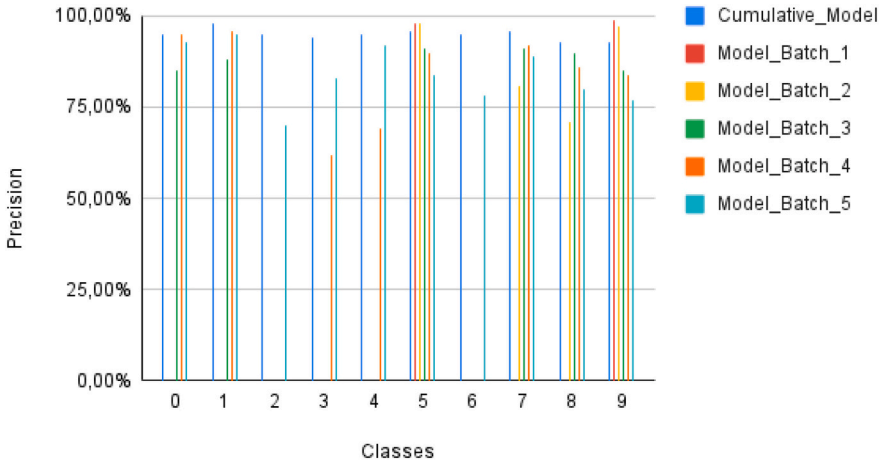

Fig. 14. Performance comparison on the MNIST dataset. The model $Model\_Batch_{-1}$ (red) learns the classes"5″ and"9″. The model $Model\_Batch_{-2}$ (yellow) learns the classes"7″ and"8″. The model $Model\_Batch_3$ (green) learns the classes"0″ and"1″. Last the models $Model\_Batch_4$ (orange) and $Model\_Batch_5$ (cyan) learn the classes"3″,"4″ and"2″,"6″ respectively. Whereas, the Cumulative Model (blue) learns the classes in the same order relying on the entire dataset. The precision of the class"4″ achieved by the $Model\_Batch_4$ increases using the Model Batch 5 from 69 % to 92 %.

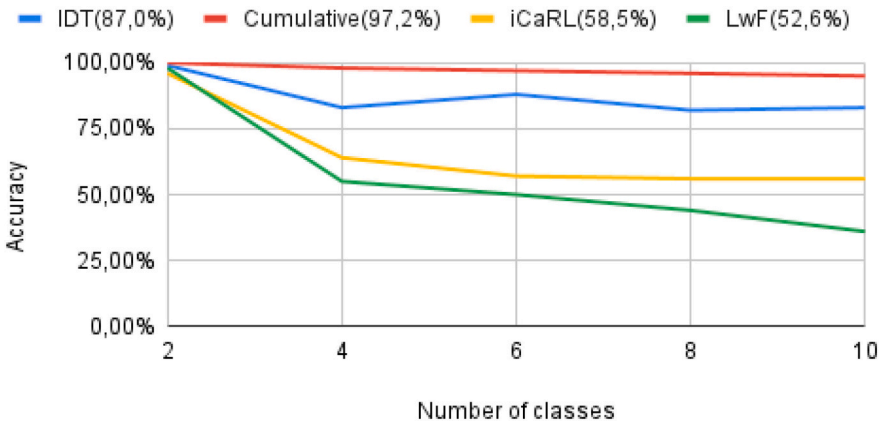

Fig. 15. Performance comparison on MNIST dataset. After learning the 10 classes, the IDT (blue) achieved 83,2 % against only 56,1 % and 36,0 % for iCaRL and LwF respectively. For each model, the average accuracy is reported between brackets. The Cumulative model is trained on the entire dataset as an upper bound experimentation.

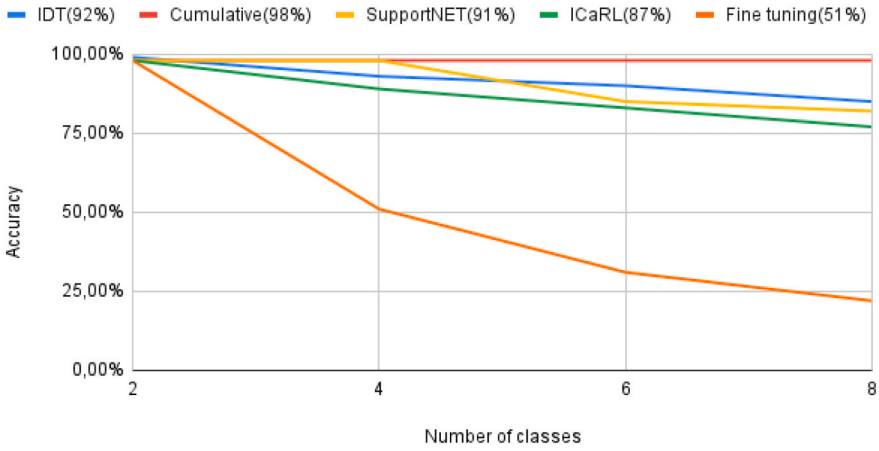

Fig. 16. Performance comparison on the BreakHis dataset. After learning 4 classes, the SupportNet (yellow) achieved +5 % than the IDT (blue). However, our model performed better with +5 % and + 3 % after learning the remaining classes. For each model, the average accuracy is reported between brackets. The Cumulative model is trained on the entire dataset as an upper bound experimentation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

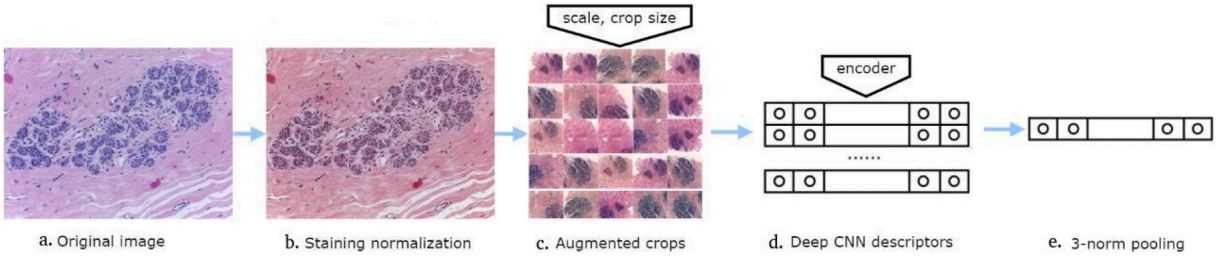

Fig. 17. Image augmentation and feature extraction. The original RGB image 700 × 460 pixels is presented in (a). From the normalized staining (b), random crops of 400 × 400 pixels are extracted (c) and encoded into descriptors (d) then pooled into a single descriptor (e).

and *Model_Batch*$_2$ increase, using the *Model_Batch*$_5$, from 88 %, 62 % and 81 % to 96 %, 83 % and 91 % respectively.

Thus, updating the IDT knowledge allows it to improve its skills on old classes. The *Cumulative_Model* refers to an upper bound experimentation which learns the classes when new batches of data are available relying on the entire dataset (old and new).

Further, the blue curve in Fig. 15 reports the accuracy achieved by the IDT which reaches about 88 % in the last incremental session. The average accuracy, reported between brackets, reaches 87,0 % exceeding both the iCaRL (58,5 %) in yellow and the LwF (52,6 %) in green. This is due to the fact that the LwF does not use a rehearsal memory for old tasks, while the iCaRL relies mainly on the representation quality of the replayed data.

The above findings are affirmed by the diagonally dense confusion matrix, presented in Fig. 12, which reports the IDT class predictions performed after learning the last batch of data (*Batch_5*).

5.3.2. *BreakHis dataset*

As mentioned in the Section 5.1, the BreakHis dataset images are 3-channel RGB with 50 pixels. For each one of them, 50 random colour augmentations are performed, and the magnitude of every pixel is multiplied by two random uniform variables from the range [0.7, 1.3] as suggested in [3].

After that, from each original image, 20 random crops of 400 × 400 pixels are extracted and then encoded into 50 descriptors which are combined and pooled into a single descriptor of the image using $p = 3$

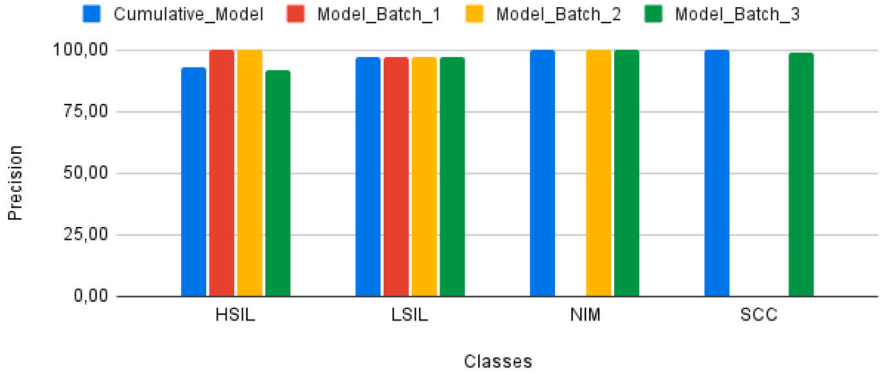

Fig. 18. Performance comparison on the LBC dataset. The Cumulative Model (blue) is trained on the entire dataset as an upper bound model. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

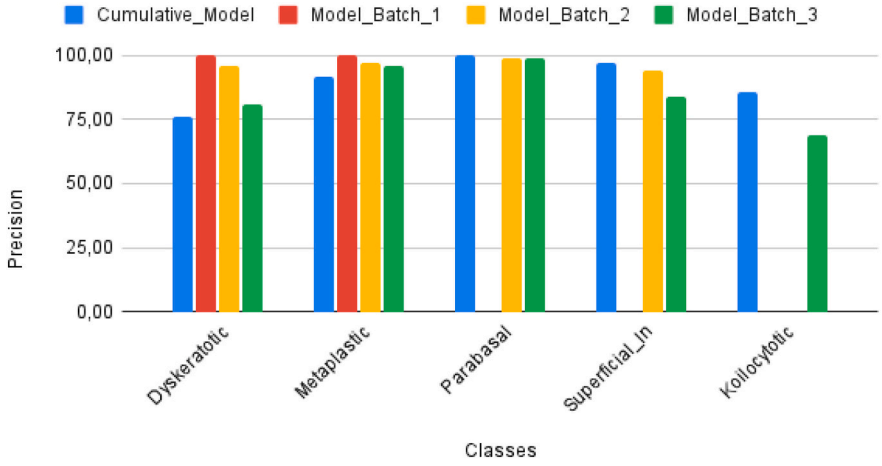

Fig. 19. Performance comparison on the SIPaKMeD dataset. The Cumulative Model (blue) is trained on the entire dataset as an upper bound model. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

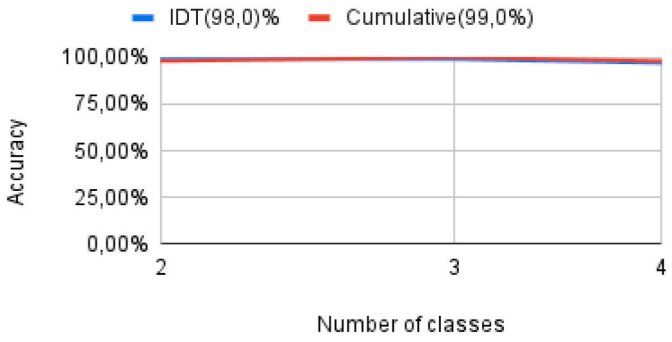

Fig. 20. Performance comparison on the LBC dataset.

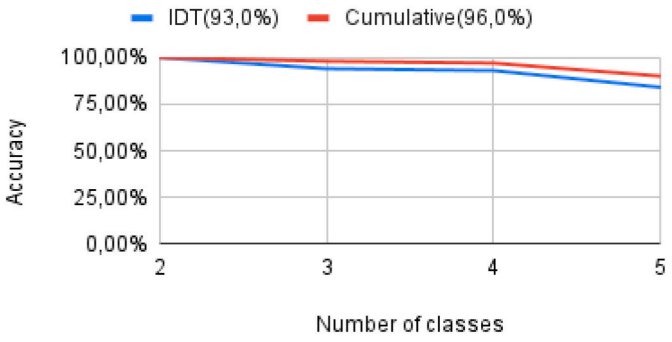

Fig. 21. Performance comparison on the SIPaKMeD dataset.

(see Section 4.1.1). Accordingly, 50 augmentations are obtained for each original image as shown in Fig. 17.

Further, the dataset is divided into three subsets. 20 % of the original data is kept for tests, whereas 10 % of the training set is used for the validation. The experimentations are conducted using 4 batches with 2 different classes in each one of them.

The obtained results show that the IDT exceeds all the compared methods (LwF, iCaRL, SupportNet and *fine-tuning*), excepting the case of the second batch, where the recorded accuracy is 93 % against 98 % achieved by the SupportNet (see Fig. 16). However, after learning the 3rd and the 4th batches, our method achieved 90 % and 85 % exceeding the SupportNet (yellow) by 5 % and 3 % respectively.

The *all data configuration* (red) refers to an upper boundary experimentation for learning without forgetting. In contrast, the *fine-tuning* experimentation (orange), shows the impact of catastrophic forgetting with the increase of the number of classes.

For the average accuracies, our method achieves 92,0 % against 91 %, 87 % and 51 % for SupportNet, iCaRL and *fine-tuning* respectively. This is explained by the fact that *fine-tuning* deep models give rise to a high plasticity leading to catastrophic forgetting. Moreover, the iCaRL degrades drastically on bioinformatics datasets as demonstrated in [18]. Thus, the proposed method is able to achieve acceptable predictions as shown in the confusion matrix (see Fig. 13).

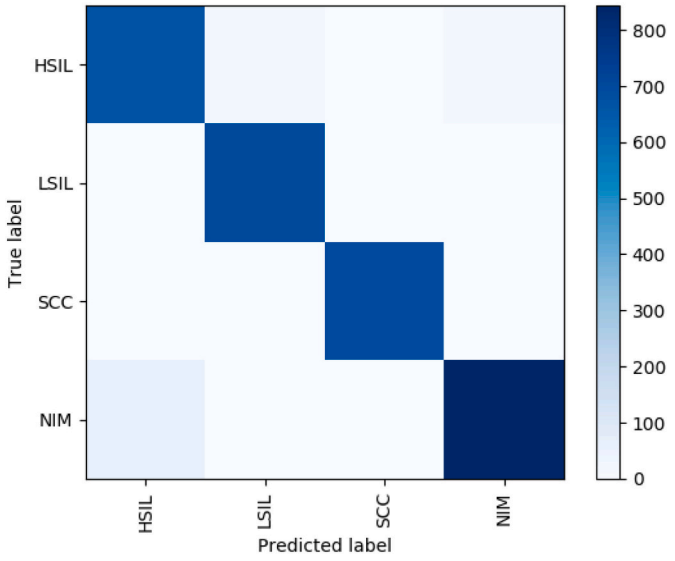

Fig. 22. The IDT confusion matrix on the LBC dataset.

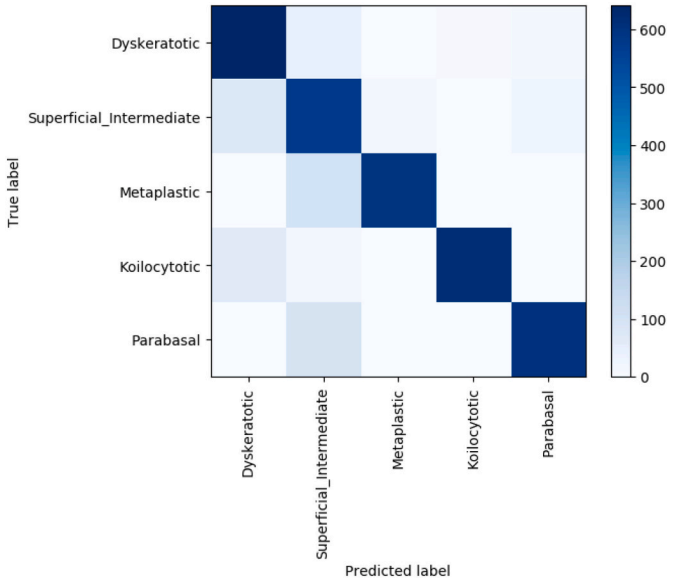

Fig. 23. The IDT confusion matrix on the SIPaKMeD dataset.

5.3.3. Pap smear datasets

Referring to the algorithm presented in the Section [par: Learning the first batch], the first batch of data must contain at least two classes. As the LBC and the SIPaKMeD datasets contain 4 and 5 classes respectively, the framework learns two classes in the first batch of data, then a new class is added per batch in the next learning sessions (rounds).

At the end of the incremental learning on the LBC dataset, the obtained results show that the *"HSIL"* class precision slightly decreases after learning the *"NIM"* and the *"SCC"* classes. However, the IDT and the *all data configuration* achieved the same precision 92 % as shown in Fig. 18. In contrast, the class precisions for *"LSIL"* and *"NIM"* are greater than or equal to the obtained values with the *all data configuration* (blue bars for the *cumulative model*).

The same findings are drawn for the *"Dyskeratotic"* and the *"Metaplastic"* classes from the SIPaKMeD dataset as shown in the Fig. 19, in which the cumulative bars in blue refer to the *all data configuration* as an upper bound experimentation.

In addition, after learning 5 classes from the SIPaKMeD dataset, the IDT achieved 6 % less accuracy than the *all data configuration* (90 %). However, the average accuracy reached 93 % against 96 % in the upper bound experimentation which represents only 3 % of difference.

For the LBC dataset, the obtained curves are superimposed which proves that the IDT is as efficient as the *all data configuration* with 97 % accuracy after learning all the classes. The average accuracy in Fig. 20 is 98 % for the LBC dataset. This could be explained by the fact that the cytological images contain single cells, in contrast with the SIPaKMeD images which contain tissues of cells.

Finally, the performance of the IDT classification is summarised in Fig. 22 and Fig. 23, where the obtained results are promising since the confusion matrices are diagonally dense.

Unlike the BreakHis dataset which has been incrementally experimented using the SupportNet framework, our paper is, to the best of our knowledge, the first work that addresses the incremental learning in cervical cancer. So, comparative studies with the Tree-CNN and the ACLSeg were not carried out. This is due to the fact that, on one hand the target metrics in ACLSeg are the *dice coefficient* and the *knowledge retention*. On the other hand, when the Tree-CNN updates its knowledge, it relies on the new batches of data jointly with old ones, which is out of the scope of this paper.

6. Conclusion

In this work we present a new *Incremental Deep Tree framework* entitled the IDT, which allows a deep convolutional neural network to maintain previous skills while acquiring new knowledge. The proposed model performs prediction of new classes while achieving acceptable accuracies on the old ones. Actually, it relies on two principles: replay a part of old data when learning new classes and for each previous task, a target output is included in the new model. The former is devoted to support the features from old classes, while the latter aims to mitigate the forgetting of these features. Unlike most of the continual learning algorithms, the proposed method combines the three strategies of continual learning namely the *replay*, the *regularization* and the *parameter-isolation* based methods.

The IDT was experimented on the MNIST and three biological datasets. The results on the BreakHis, the LBC, the SIPaKMeD and the MNIST datasets are promising as they exceed the studied methods of the state-of-the-art. As biological datasets are difficult to access and they are regularly growing, the proposed framework helps specialists in the field of continual learning for Computer-Aided-Diagnosis.

This work opens up avenues for research on the architecture of the *base model*. As the proposed method acts exclusively on the output layer of the classifier and does not focus on its inner architecture, investigating in new deep models with a hyper-parameter optimisation would improve the obtained results. Further, the achieved precisions could be increased by enhancing the quality of the replayed images using machine learning techniques like support vector machines or generative adversarial networks to produce synthetic images. Such a flexibility consequently enables the application of the IDT framework on other types of images and in other fields.

Funding

No funding was received for this work.

Declaration of competing interest

No conflict of interest exists.

Acknowledgments

We are grateful to the Algerian Ministry of Higher Education and Scientific Research (MESRS) and the Algerian General Directorate of Scientific Research and Technological Development (DGRSDT) for the

financial support. This work was granted access to the HPC resources of UCI-UFMC (Unité de Calcul intensif of the University FRERES MENTOURI CONSTANTINE1).

References

[1] World Health Organisation. Breast cancer. 2020.
[2] World Health Organisation. Cervical cancer. 2020.
[3] Rakhlin A, Shvets A, Iglovikov V, Kalinin AA. Deep convolutional neural networks for breast cancer histology image analysis. 2018.
[4] Kaur I, Doja MN, Ahmad T. Data mining and machine learning in cancer survival research: an overview and future recommendations. J Biomed Inform 2022;128:104026. https://doi.org/10.1016/j.jbi.2022.104026. Disponible sur.
[5] Chu T, Wang J, Chen J. An adaptive online learning framework for practical breast cancer diagnosis. In: Medical imaging 2016: computer-aided diagnosis. [s.l.]: SPIE; 2016. p. 537–48.
[6] Maicas G, Carneiro G, Bradley AP, Nascimento JC, Reid I. Deep reinforcement learning for active breast lesion detection from DCE-MRI. In: International conference on medical image computing and computer-assisted intervention. [s.l.]: Springer; 2017. p. 665–73.
[7] Shen C, Gonzalez Y, Jung H, Chen L, Qin N, Jia X. Automatic inverse treatment planning for cervical cancer high dose-rate brachytherapy via deep reinforcement learning. Int J Radiat Oncol Biol Phys 2018;102(3):e540.
[8] Navarro F, Sekuboyina A, Waldmannstetter D, Peeken JC, Combs SE, MenzeB H. Deep reinforcement learning for organ localization in CT. In: Medical imaging with deep learning. [s.l.]: PMLR; 2020. p. 544–54.
[9] Coronato A, Naeem M, de Pietro G, Paragliola G. Reinforcement learning for intelligent healthcare applications: a survey. Artif Intell Med 2020;109:101964.
[10] Fu L, Xia W, Shi W, Cao G, Ruan Y, Zhao X, Liu M, Niu S, Li F, Gao X. Deep learning based cervical screening by the cross-modal integration of colposcopy, cytology, and HPV test. Int J Med Inform 2022;159:104675.
[11] Kashyap H, Ahmed HA, Hoque N, Roy S, BhattacharyyaD K. Big data analytics in bioinformatics: architectures, techniques, tools and issues. NetwModelAnalHealth InfBioinform 2016;5(1):28.
[12] Abhishek P, Ramesh V. A survey on deep leaning architectures and its applications. IntJEducSci 2018.
[13] Hussain E, Mahanta LB, Das CR, Choudhury M, Chowdhury M. A shape context fully convolutional neural network for segmentation and classification of cervical nuclei in Pap smear images. ArtifIntellMed 2020;107:101897.
[14] Mat-Isa NA, Mashor MY, Othman NH. An automated cervical pre-cancerous diagnostic system. ArtifIntellMed 2008;42(1):1–11.
[15] Abd-Elnaby M, Alfonse M, Roushdy M. Classification of breast cancer using microarray gene expression data: a survey. J Biomed Inform 2021;117:103764.
[16] Jieun K, Youngno Y, Sungwon K, Kyunghwa H, Eun-Kyung K. Deep learning for the detection of breast cancers on chest computed tomography. Clin Breast Cancer 2022;22(1):26–31.
[17] Luuk B, Jonas T, Ritse MM. Application of deep learning in breast cancer imaging. Semin Nucl Med 2022;52(5):584–96.
[18] Li Y, Li Z, Ding L, Hu Y, Chen W, Gao X. SupportNet: a novel incremental learning framework through deep learning and support data. bioRxivBioinform 2018.
[19] de Lange M, Aljundi R, Masana M, Parisot S, Jia X, Leonardis A, Slabaugh G, Tuytelaars T. Continual learning: a comparative study on how to defy forgetting in classification tasks. 2019.
[20] Shmelkov K, Schmid C, Alahari K. Incremental learning of object detectors without catastrophic forgetting. In: Proceedings of the IEEE International Conference on Computer Vision. [s.l.]: [s.n.]; 2017. p. 3400–9.
[21] Ven GMvan de, Tolias AS. Three scenarios for continual learning. 2019.
[22] Bagus B, Gepperth A. An investigation of replay-based approaches for continual learning. In: 2021 International Joint Conference on Neural Networks (IJCNN). [s.l.]: IEEE; 2021. p. 1–9.
[23] Ching T, Himmelstein DS, Beaulieu-Jones BK, Kalinin AA, Do BT, Way GP, Ferrero E, Agapow P-M, Zietz M, Hoffman MM, et al. Opportunities and obstacles for deep learning in biology and medicine. J R Soc Interface 2018;15(141):20170387.
[24] Rebuffi S-A, Kolesnikov A, Sperl G, Lampert CH. icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. [s.l.]: [s.n.]; 2017. p. 2001–10.
[25] Rolnick D, Ahuja A, Schwarz J, Lillicrap TP, Wayne G. Experience replay for continual learning. 2018.
[26] Shin H, Lee JK, Kim J, Kim J. Continual learning with deep generative replay. In: Advances in neural information processing systems. [s.l.]: [s.n.]; 2017. p. 2990–9.
[27] Elskhawy A, Lisowska A, Keicher M, Henry J, Thomson P, Navab N. Continual class incremental learning for CT thoracic segmentation. 2020.
[28] Ebrahimi S, Meier F, Calandra R, Darrell T, Rohrbach M. Adversarial continual learning. 2020.
[29] Lopez-Paz D, Ranzato M. Gradient episodic memory for continual learning. In: Advances in neural information processing systems. 30; 2017. p. 6467–76.
[30] Aljundi R, Lin M, Goujaud B, Bengio Y. Online continual learning with no task boundaries ». arXiv preprint arXiv:1903.08671. 2019.
[31] Li Z, Hoiem D. Learning without forgetting. IEEE Trans Pattern Anal Mach Intell 2017;40(12):2935–47.
[32] Zhang J, Zhang J, Ghosh S, Li D, Tasci S, Heck L, Zhang H, Kuo C-CJ. Class-incremental learning via deep model consolidation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision; 2020. p. 1131–40 [s.l.]: [s.n.].
[33] Huszár F. On quadratic penalties in elastic weight consolidation. 2017. arXiv preprint arXiv:1712.03847.
[34] Schwarz J, Czarnecki W, Luketina J, Grabska-Barwinska A, Teh YW, Pascanu R, Hadsell R. Progress & compress: a scalable framework for continual learning. In: International Conference on Machine Learning. [s.l.]: PMLR; 2018. p. 4528–37.
[35] Aljundi R, Babiloni F, Elhoseiny M, Rohrbach M, Tuytelaars T. Memory aware synapses: learning what (not) to forget. In: Proceedings of the European Conference on Computer Vision (ECCV). [s.l.]: [s.n.]; 2018. p. 139–54.
[36] Mao F, Weng W, Pratama M, Yee EYK. Continual learning via inter-task synaptic mapping [En ligne] Knowl-Based Syst 2021;222:106947. https://doi.org/10.1016/j.knosys.2021.106947. Disponible sur.
[37] Li X, Zhou Y, Wu T, Socher R, Xiong C. Learn to grow: a continual structure learning framework for overcoming catastrophic forgetting. In: International Conference on Machine Learning. [s.l.]: PMLR; 2019. p. 3925–34.
[38] Roy D, Panda P, Roy K. Tree-CNN: a hierarchical deep convolutional neural network for incremental learning. Neural Netw 2020;121:148–60.
[39] Rusu AA, Rabinowitz NC, Desjardins G, Soyer H, Kirkpatrick J, Kavukcuoglu K, Pascanu R, Hadsell R. Progressive neural networks. 2016.
[40] Pratama M, Ashfahani A, Lughofer E. Unsupervised continual learning via self-adaptive deep clustering approach. 2021.
[41] Chen Z, Liu B. In: Lifelong machine learning. Synthesis lectures on artificial intelligence and machine learning, 12(3); 2018. p. 1–207.
[42] Chollet F. Deep learning with Python. [s.l.]: Manning Publications; 2018.
[43] Mousser W, Ouadfel S. Deep feature extraction for pap-smear image classification: a comparative study. In: Proceedings of the 2019 5th International Conference on Computer and Technology Applications. [s.l.]: [s.n.]; 2019. p. 6–10.
[44] Mermillod M, Bugaiska A, Bonin P. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects. Front Psychol 2013;4:504.
[45] Lecun Yann, Cortes CJCBCorinna. THE MNIST DATABASE of handwritten digits [En ligne]. Disponible sur, http://yann.lecun.com/exdb/mnist/; 2012.
[46] Grother PJ. NIST special database 19. In: Handprinted forms and characters database. National Institute of Standards and Technology; 1995. p. 10.
[47] Hussain E, Mahanta LB, Borah H, Das CR. Liquid based-cytology Pap smear dataset for automated multi-class diagnosis of pre-cancerous and cervical cancer lesions. Data Brief 2020;30:105589.
[48] Plissiti ME, Dimitrakopoulos P, Sfikas G, Nikou C, Krikoni O, Charchanti A, SIPAKMED check. A new dataset for feature and image based classification of normal and pathological cervical cells in Pap smear images. In: 2018 25th IEEE International Conference on Image Processing (ICIP). [s.l.]: IEEE; 2018. p. 3144–8.
[49] Chollet F, et al. Keras MNiST ConvNet. [s.l.]: GitHub; 2015. https://github.com/keras-team/keras-io/blob/master/examples/vision/mnist_convnet.py.