Continual learning for surface defect segmentation by subnetwork creation and selection

Dekhovich, Aleksandr; Bessa, Miguel A.

DOI
10.1007/s10845-024-02393-4

Publication date
2024

Document Version
Final published version

Published in
Journal of Intelligent Manufacturing

Citation (APA)
Dekhovich, A., & Bessa, M. A. (2024). Continual learning for surface defect segmentation by subnetwork creation and selection. *Journal of Intelligent Manufacturing*, *36 (2025)*(5), 3051-3065. https://doi.org/10.1007/s10845-024-02393-4

Journal of Intelligent Manufacturing
https://doi.org/10.1007/s10845-024-02393-4

Continual learning for surface defect segmentation by subnetwork creation and selection

Aleksandr Dekhovich[1] · Miguel A. Bessa[2]

Received: 6 December 2023 / Accepted: 30 March 2024

Abstract

We introduce a new continual (or lifelong) learning algorithm called LDA-CP&S that performs segmentation tasks without undergoing catastrophic forgetting. The method is applied to two different surface defect segmentation problems that are learned incrementally, i.e., providing data about one type of defect at a time, while still being capable of predicting every defect that was seen previously. Our method creates a defect-related subnetwork for each defect type via iterative pruning and trains a classifier based on linear discriminant analysis (LDA). At the inference stage, we first predict the defect type with LDA and then predict the surface defects using the selected subnetwork. We compare our method with other continual learning methods showing a significant improvement – mean Intersection over Union better by a factor of two when compared to existing methods on both datasets. Importantly, our approach shows comparable results with joint training when all the training data (all defects) are seen simultaneously.

Keywords Continual learning · Automatic vision inspection · Surface defect segmentation · Linear Discriminant Analysis (LDA)

Introduction

Automatic defects inspection plays an important role in product quality evaluation (Prunella et al., 2023). In the beginning of the field, the creation of meaningful features to find defective regions was done manually (Ojala et al., 2002; Chao and Tsai, 2008; Song and Yan, 2013; Jeon et al., 2014). Although classical machine learning methods have been proposed to identify images with defective surfaces (Jia et al., 2004; Agarwal et al., 2011; Shanmugamani et al., 2015), recent advances in deep learning research have led to an increase in performance (Prunella et al., 2023). Typically, there are three types of tasks for defect inspection with neural networks – classification, detection (He et al., 2019) and segmentation (Tabernik et al., 2020). In the case of defect classification, transfer learning helps to increase the network's ability to detect defective surfaces (Aslam et al., 2020; Wu and Lv, 2021). For segmentation, most methods are based on the U-Net architecture (Ronneberger et al., 2015) taking advantage of convolutional layers that automatically extract features from the images of the surfaces (He et al., 2019; Song et al., 2020; Hao et al., 2021; Huang et al., 2020). Attention mechanisms (Vaswani et al., 2017) employed in the model's architecture can lead to even more accurate predictions (Pan and Zhang, 2022; Üzen et al., 2022).

The advent of deep learning models came with more data for training and comparing these models in different real-life scenarios. For instance, after Song and Yan (2013) proposed their NEU-DET dataset with Hot Rolled Steel Strip Surface defects, containing six types of defects, other groups collected datasets with either different defect categories or a more significant number of defects, e.g., GC10-DET (Lv et al., 2020) and X-SDD (Feng et al., 2021). In segmentation literature, we can also find examples of different categorizations of surface defects, e.g., the Magnetic tile dataset (Huang et al., 2020) contains images of five types of defects together with defect-free cases. As a final example, the dataset collected by Liu and Ye (2022) also contains a large number of images but only has three types of defects.

✉ Miguel A. Bessa
miguel_bessa@brown.edu

[1] Department of Materials Science and Engineering, Delft University of Technology, Mekelweg 2, Delft 2628 CD, The Netherlands

[2] School of Engineering, Brown University, 184 Hope St., Providence RI 02912, USA

Published online: 06 May 2024

Notwithstanding the increase in availability of datasets, there are many instances where there are few types of defects in each dataset. This is a natural occurrence in Engineering practice because many processes are not amenable to high-throughput. Simultaneously, if new defects occur or if another defect identification task with similar characteristics is encountered, using the original dataset and neural network model while considering new types of defects in similar (or even different) materials can be invaluable. However, training the same neural network model on a new dataset currently requires retraining it on all the data, even if the model was already capable of detecting some types of defects. This happens because deep learning models suffer from catastrophic forgetting (French, 1999; Goodfellow et al., 2014; Coop et al., 2013). In conventional training, neural networks cannot learn new tasks without forgetting old ones if the tasks are learned incrementally. Instead, the continual learning field (De Lange et al., 2021) aims to solve this type of problem where the model receives data in batches (tasks) but aims to learn information mitigating the forgetting issues.

We illustrate the impact of catastrophic forgetting on segmentation tasks in Fig. 1 by considering the defect segmentation dataset SD-saliency-900 (Song et al., 2020). This dataset consists of images with three types of defects: scratches, patches and inclusions. We illustrate this phenomenon by focusing on three typical learning scenarios: 1) *single-task* training where each defect is learned with a single network, meaning there are three networks in total; 2) *joint* training where the model has access to the entire dataset at once; 3) *finetuning*, in which the network learns to segment sequentially, adapting the parameters for the new task, having them pretrained on previous ones.

For all three learning scenarios, we quantify the segmentation performance via the mean Intersection over Union (mIoU) score for every task after each incremental step (Fig. 1). We observe that finetuning on a new task leads to a significant drop in performance for the previous task(s), as indicated by the blue bars – a clear illustration that learning a sequence of tasks with a single network leads to forgetting the previous tasks in the sequence (catastrophic forgetting). However, forgetting does not occur in the case of single- and joint-task training because the network is capable of learning each of the defects separately without any pretraining, while also being capable of learning all of them together. We also note that both single- and joint-task training have comparable performance, despite a small decrease in the latter case.[1]

Therefore, we see that the ability to predict defects of previous types is lost when training for a new type of defect, i.e. that is out-of-distribution. The main objective of our work is to propose a continual learning algorithm suitable for the surface segmentation problem. Therefore, we developed a novel continual learning algorithm that performs significantly better than the state of the art: more than two times better according to two different segmentation performance metrics and for two different datasets. Moreover, the proposed approach does not exhibit any forgetting and is comparable with joint training, i.e. when all the data is used for training (no continual learning). To the best of our knowledge, this is the first work that addresses the catastrophic forgetting issue and develops a continual learning approach for surface defect segmentation.

The article is organized as follows: "Continual learning" Section gives a brief overview of continual learning methods and their application in manufacturing processes, "Proposed approach" Section describes the proposed approach, "Experiments and results" Section provides numerical results and illustrates the comparison with other continual learning algorithms, and "Conclusion" Section summarizes the work.

Continual learning

Overcoming the above-mentioned catastrophic forgetting requires deep learning models to be trained in a continual (or lifelong) learning manner (Thrun and Pratt, 1998). The overwhelming majority of continual learning literature is dedicated to classification tasks. In that context, three different categories have emerged (De Lange et al., 2021): regularization-based (Li and Hoiem, 2017; Zenke et al., 2017; Aljundi et al., 2018), replay-based (Rebuffi et al., 2017; Castro et al., 2018; He et al., 2019; Douillard et al., 2020) and architectural-based methods (Mallya and Lazebnik, 2018; Sokar et al., 2022; Dekhovich et al., 2023). In some cases, there are methods that can fall into more than one category (Yan et al., 2021; Wang et al., 2022). Often these methods show better performance but require more memory or have high computational cost (e.g., extra memory buffer or architecture extension), which creates specific challenges when deployed in real-life applications.

Regularization-based methods penalize parameters obtained on incremental step $t-1$ from drastic changes while learning the task on incremental step t. For example, SI (Zenke et al., 2017), EWC (Kirkpatrick et al., 2017) and MAS (Aljundi et al., 2018) employ total loss $\mathcal{L}^{(t)}(x;\theta^{(t)})$ on incremental step t that consists of the loss computed for the current data, and a penalty term to prevent forgetting:

[1] As a short note, marginal improvements in performance sometimes occur when changing the task order (investigated at the end of the article). For example, the mIoU performance for the Scratches task improved by 0.13 points after learning the Inclusion task, but the improvement is small compared to how much it degrades after learning the Patches task.

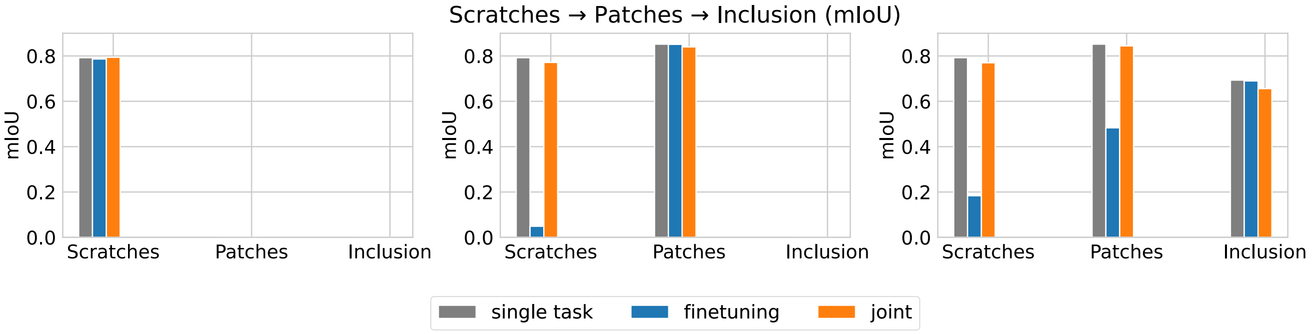

Fig. 1 Example of forgetting in the case of incremental learning of three types of defects

$$\mathcal{L}^{(t)}\left(x;\theta^{(t)}\right) = \mathcal{L}_{curr}\left(x;\theta^{(t)}\right) + \frac{\lambda}{2}\sum_{i=1}^{\#params}\Omega_i\left(\theta_i^{(t)} - \theta_i^{(t-1)}\right)^2, \qquad (1)$$

where $\mathcal{L}_{curr}\left(x;\theta^{(t)}\right)$ is a loss on the current data, $\sum_{i=1}^{\#params}\Omega_i\left(\theta_i^{(t)} - \theta_i^{(t-1)}\right)^2$ is the penalty term, Ω_i is the cumulative importance for parameter i, and $\theta^{(t-1)}, \theta^{(t)}$ are network parameters at incremental steps $t-1$ and t, respectively. Learning without forgetting (LwF) (Li and Hoiem, 2017) aims to mitigate forgetting by minimizing the cross-entropy between output probabilities before and after the model is trained on a new task.

Replay-based (or rehearsal-based) approaches use a small fraction of data from previous tasks and keep it in a fixed-size memory buffer. However, storing old data in the buffer may not be allowed due to privacy issues (Zhang et al., 2020). Also, if the model parameters were downloaded without the memory buffer, further model training is not possible without forgetting. Therefore, in this work, we do not focus on this type of methods.

Architectural approaches do manipulations with the network structure by freezing and assigning some parameters to a specific task (e.g., PackNet (Mallya and Lazebnik, 2018)) or constantly growing the architecture increasing the expressivity of the network (e.g., DEN (Yoon et al., 2018)). However, if the model grows while learning a new task, the final number of parameters is not bounded, leading to additional computational costs. Alternatively, if the algorithm finds task-specific parameters, e.g., via iterative pruning in CP&S (Dekhovich et al., 2023) or pruning at initialization in SupSup (Wortsman et al., 2020), the challenge lies on activating the correct subnetwork during inference. This subnetwork selection in both CP&S and SupSup requires a batch of test data of the task to be predicted, such that the correct subnetwork (i.e. task ID) is identified. This may also be impractical in real-life cases.

Literature on continual learning for semantic segmentation is scarcer. We can find examples that adapt classification continual learning algorithms to segmentation (Baweja et al., 2018; van Garderen et al., 2019), or some new approaches designed specifically for segmentation (Klingner et al., 2020; Douillard et al., 2021; Yan et al., 2021). Similar to the classification case, better results are achieved by the methods that use a fixed-size memory buffer with samples from old tasks to overcome forgetting (Cha et al., 2021; Qiu et al., 2023). However, even though these methods use old data (facilitating training), they still show significant forgetting of the first tasks while performing well only on the last ones.

Continual learning also finds its application in industrial and manufacturing cases. For example, MAS (Aljundi et al., 2018) was applied for product quality evaluation (Tercan et al., 2022). The approach clones the output head for previous tasks with the lowest loss on the current data and uses this copy as initialization for a new task. The weight transfer for the output layer, and MAS algorithm that penalizes parameters from previous layers, show good performance for the considered regression problem. Regularization-based methods have been examined for anomaly detection in manufacturing process (Maschler et al., 2021) and fault prediction in lithium-ion batteries (Maschler et al., 2022). Sun et al. (2023) developed an adaptive classification framework based on continual learning to identify new unlabeled samples. The proposed approach uses Mahalanobis distance and is employed to decide whether a new batch of data belongs to the already seen defect type, or forms a new one.

Proposed approach

We propose to take advantage of architectural methods that create task-specific subnetworks for each task, eliminating the subnetwork selection issue. As a base method, we consider Continual Prune-and-Select (CP&S) (Dekhovich et al., 2023) where we improve the subnetwork selection process by training a model for this purpose, instead of having simple metric-based decision rules. In general, the task-prediction problem is quite challenging in continual learning (Kim et al., 2020) and can be seen as an out-of-distribution (OOD) detection problem (Kim et al., 2022). The difficulty arises from the presence of arbitrary classes in each task, leading to cases where classes within each task may not be similar, while classes from different tasks may have important similarities. This poses a challenge to identify the task ID and corresponding subnetwork, affecting the performance of the continual learning model when the wrong subnetwork is selected. Conversely, these methods have the advantage that when the correct subnetwork is identified then there is no forgetting, which explains their state-of-the-art performance in different image-classification datasets (Dekhovich et al., 2023).

However, in contrast to image classification, every task in defect segmentation problems consists of defects of only one type. This represents an opportunity for architectural continual learning methods because we can train a model that learns the distribution of each defect separately. To do so, we use linear discriminant analysis trained on features extracted from a pretrained convolutional neural network (Dorfer et al., 2016; Hayes and Kanan, 2020). For the segmentation model, we use the U-Net architecture (Ronneberger et al., 2015), in which we create task-specific subnetworks via iterative pruning. As a pretrained feature extractor, we use the EfficientNet-B5 architecture (Tan and Le, 2019) pretrained on ImageNet-1000 (Deng et al., 2009).

In Fig. 2, we illustrate the inference stage of our approach, which consists of two steps: (1) predicting the defect type (task ID) with LDA; and (2) using a subnetwork that corresponds to the predicted defect to predict the segmentation mask. Note that at the inference stage, defect type prediction and defect mask prediction need to be done sequentially. Training for these steps can be done in parallel and independently from each other. We call our proposed approach LDA-CP&S since it uses the CP&S paradigm of creating subnetworks during training, and it employs LDA for the subnetwork selection.

Referring again to Fig. 1, we recall that the three separate models (single-task grey bars) are capable of learning the defects slightly better than joint training with all the tasks together (orange bars). This hints that having task-specific parameters associated with only one task can even help the learning process. At the same time, the shared parameters provide a transfer learning effect between a new subnetwork and all the ones created before. Both advantages can be exploited by LDA-CP&S.

Notwithstanding, our method could suffer a performance drop from two possible sources: the pruning stage, and the LDA classification stage. The performance reduction due to pruning may occur because some important parameters could be deleted when creating additional space (free connections) for future tasks. In addition, misclassification by LDA could result in signal routing through the wrong subnetwork and consequently poor segmentation performance. In "Experiments and results" Section, we show that these two sources of error are negligible compared to the benefits of our approach. In the following subsections, we describe the processes for subnetwork creation and LDA training.

Subnetwork creation

To create a subnetwork for the given task, we use NNrelief pruning algorithm (Dekhovich et al., 2024). The approach evaluates the strength of the signal that propagates through every connection/kernel. This pruning technique shows better sparsity results than other connection/kernel-based pruning techniques (Han et al., 2015; Li et al., 2017; Lee et al., 2019).

For the set of m_{l-1}-channelled input samples $\mathbf{X}^{l-1} = \{\mathbf{x}_1^{l-1}, \dots, \mathbf{x}_N^{l-1}\}$, where $\mathbf{x}_k^{l-1} = (x_{k1}^{l-1}, \dots, x_{km_{l-1}}^{l-1}) \in \mathbb{R}^{m_{l-1} \times h_{l-1}^1 \times h_{l-1}^2}$ with h_{l-1}^1 and h_{l-1}^2 being the height and width of feature maps for convolutional layer l. For every kernel $\mathbf{K}_{1j}^l, \mathbf{K}_{2j}^l, \dots, \mathbf{K}_{m_l j}^l$, $\mathbf{K}_{ij}^l = (k_{ijqt}^l) \in \mathbb{R}^{r_l \times r_l}$, $q \geq 1$, $r_l \geq t$, where r_l is a kernel size, and for every bias $b_j^{(l)}$ in filter $\mathbf{F}_j^l$, we define $\hat{\mathbf{K}}_{ij}^l = \left(\left|k_{ijqt}^l\right|\right)$ as a matrix consisting of the absolute values of the matrix $\mathbf{K}_{ij}^{(l)}$. Then we compute importance scores $s_{ij}^l, i \in \{1, 2, \dots, m_l\}$ of kernels $\mathbf{K}_{ij}^l$ as follows:

$$s_{ij}^l = \frac{\frac{1}{N}\sum_{n=1}^N \left|\left|\hat{\mathbf{K}}_{ij}^l * \left|x_{ni}^{l-1}\right|\right|\right|_F}{S_j^l}, \tag{2}$$

where $S_j^l = \sum_{i=1}^{m_{l-1}} \left(\frac{1}{N}\sum_{n=1}^N \left|\left|\hat{\mathbf{K}}_{ij}^l * \left|x_{ni}^{l-1}\right|\right|\right|_F\right)$ is the total importance score in filter $\mathbf{F}_j^l$ of layer l, with $*$ indicating a convolution operation, and where $||\cdot||_F$ is the Frobenius norm.

The sketch of the algorithm for pruning filter $\mathbf{F}_j^l$ in a convolutional layer l can be described as follows:

1. Choose $\alpha \in (0, 1)$ – the amount of kernels' importance that we want to keep relative to the total importance of the kernels in the filter $\mathbf{F}_j^l$.
2. Compute importance scores s_{ij}^l for all kernels in the filter $\mathbf{F}_j^l$, $i = 1, \dots, m_{l-1}$, using Eq. 2.
3. Sort importance scores s_{ij}^l for the filter $\mathbf{F}_j^l$.
4. For the sorted importance scores $\hat{s}_{ij}^l$ find minimal $p \leq m_{l-1}$ such that $\sum_{i=1}^p \hat{s}_{ij}^l \geq \alpha$.
5. Prune kernels with the importance score $s_{ij}^l < \hat{s}_{pj}^l$ for all $i \leq m_{l-1}$ and fixed j.

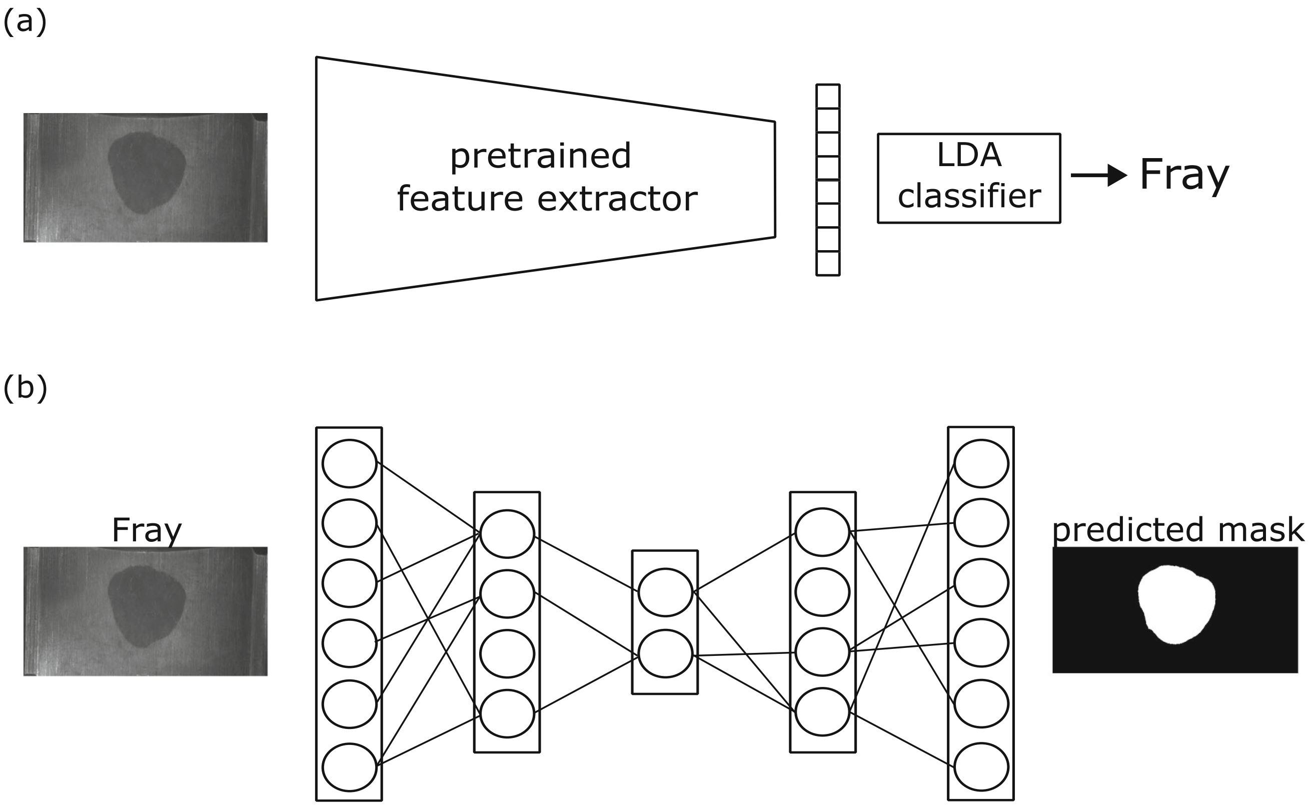

Fig. 2 An overview of the proposed method: **a** task ID (i.e., defect type/subnetwork) prediction; **b** defects segmentation

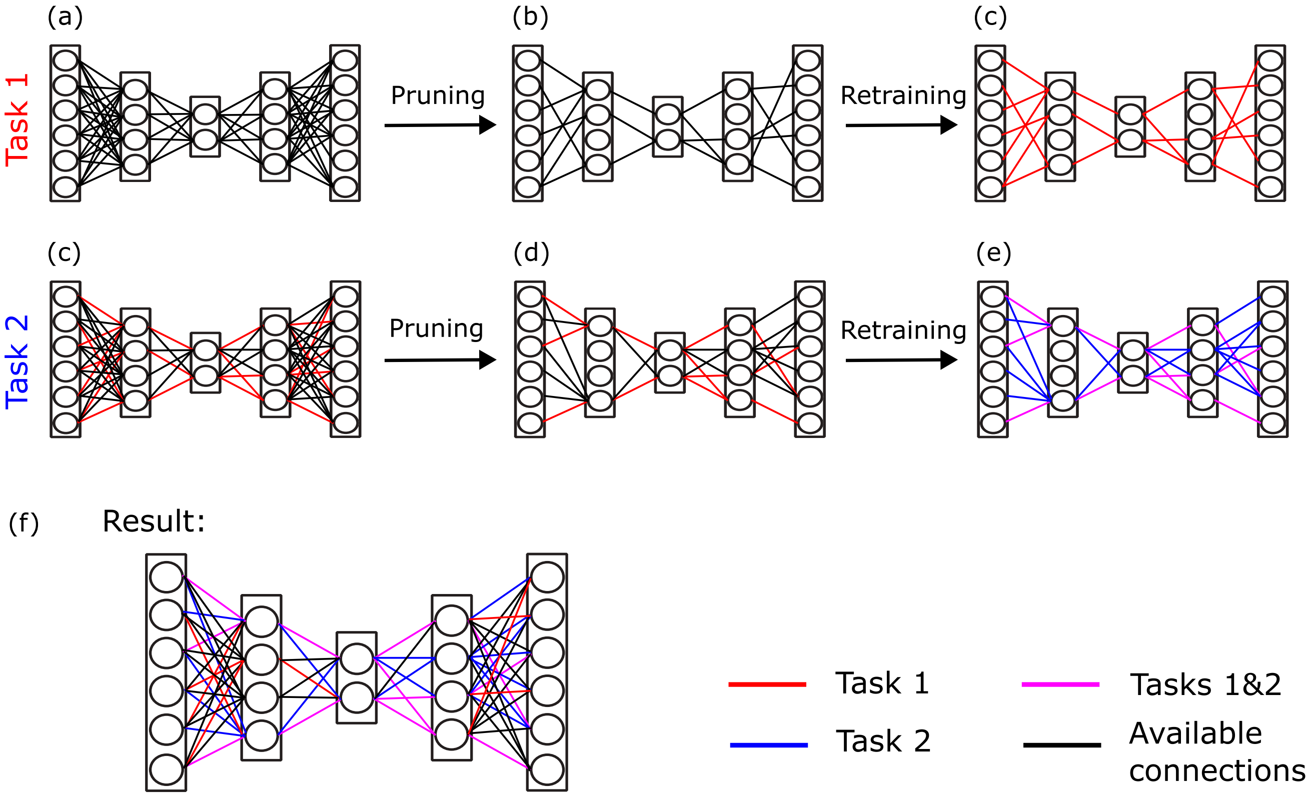

Fig. 3 An overview of Continual Prune-and-Select (CP&S): an example with two tasks

Overall, NNrelief finds kernels that propagate on average the lowest signal according to the Frobenius norm and prune these kernels. As the outcome of the procedure, we obtain a subnetwork (sub-U-Net) that predicts the defect for only one type of defects. Then we fix all parameters that are assigned to this subnetwork and do not update them anymore. When the network receives a new task with a new type of defect, CP&S finds a subnetwork for this task within the main U-Net, using the parameters assigned to the previous tasks, but without updating them. Algorithm 1 illustrates the pseudocode for CP&S and Fig. 3 illustrated the method:

Algorithm 1 Pseudocode for CP&S training procedure

```
Require: network $\mathcal{N}$, dataset $\{\mathbf{X}^{(t)}\}_{t=1}^{T}$. Initialize learning parameters (learning rate, weight decay, number of epochs, etc. ), pruning parameters (for NNrelief algorithm: $\alpha$ and $num\_iters$).
for $t = 1, 2, \ldots, T$ do
    $\mathcal{N}^{(t)} \leftarrow \mathcal{N}$
    for $iteration = 1, 2, \ldots, num\_iters$ do        ▷ repeat pruning
        $\mathcal{N}^{(t)} \leftarrow$ Pruning($\mathcal{N}^{(t)}, \mathbf{X}^{(t)}, \alpha$)        ▷ pruning step: NNrelief
        Retrain subnetworks $\mathcal{N}^{(t)}$        ▷ retraining step
    end for
    Freeze parameters $w \in \mathcal{N}^{(t)}$ and never update them
end for
Ensure: network $\mathcal{N}$ that learned tasks $1, 2, \ldots, T$.
```

We note that the proposed algorithm involves the CP&S-based subnetwork creation procedure which has complexity proportional to the number of pruning iterations. This makes the approach more costly than regularization-based ones (depending on the number of pruning iterations). However, as demonstrated next, only a few iterations are needed (between 1 and 3) and the performance improvement clearly outweighs the computational costs.

Subnetwork prediction (or selection)

To predict the task ID (type of defect) at the inference stage, we propose to use linear discriminant analysis (LDA). In this subsection, we describe the training procedure for LDA. In LDA, it is assumed that all classes have class means $\mu^{(1)}, \mu^{(2)}, \ldots, \mu^{(T)}$ and share the same covariance matrix Σ. However, in continual learning, we do not have access to all tasks at the same time, but only task t. Therefore, the covariance matrix needs to be updated online with respect to the new data batch.

Let us denote a new given task as $\mathbf{X}^{(t)} = \{x_1^{(t)}, x_2^{(t)}, \ldots, x_{N_t}^{(t)}\}$. Following streaming LDA (SLDA) strategy (Hayes and Kanan, 2020), we use a feature extractor $\mathcal{F}$ pretrained on ImageNet-1000 to obtain low-dimensional data representation $\mathbf{Z}^{(t)} := \{z_1^{(t)}, z_2^{(t)}, \ldots, z_{N_t}^{(t)}\}$, $z_i^{(t)} = \mathcal{F}(x_i^{(t)}) \in \mathbb{R}^d$. Then we can compute the class mean $\mu^{(t)} \in \mathbb{R}^d$ and update the shared covariance matrix $\Sigma^{(1:t)} \in \mathbb{R}^{d\times d}$ after incremental step t as follows (Dasgupta and Hsu, 2007):

$$\mu^{(t)} = \frac{1}{N_t}\sum_{i=1}^{N_t} z_i^{(t)} \tag{3}$$

$$\Sigma^{(1:t)} = \frac{(t-1)\Sigma^{(1:t-1)} + \Delta^{(t)}}{t}, \tag{4}$$

where $\Delta^{(t)} = \frac{(t-1)(Z^{(t)}-\mu^{(t)})(Z^{(t)}-\mu^{(t)})^{\mathsf{T}}}{t}$ and $(Z^{(t)} - \mu^{(t)}) := (z_1^{(t)} - \mu^{(t)}, z_2^{(t)} - \mu^{(t)}, \ldots, z_{N_t}^{(t)} - \mu^{(t)}) \in \mathbb{R}^{d\times N_t}$. In SLDA, the regularized version of LDA is implemented by applying shrinkage regularization to covariance matrix: $\Lambda^{(1:t)} = [(1-\varepsilon)\Sigma^{(1:t)} + \varepsilon\mathbf{I}]^{-1}$, where $\mathbf{I}$ is an identity matrix of the corresponding dimension.

At the inference stage, after learning all class means $\mu^{(t)}$, $t = 1, 2, \ldots, T$, and shared covariance matrix $\Sigma^{(1:T)}$ (and $\Lambda^{(1:t)}$ as a result), we can make a prediction for a new test sample x as follows:

$$c = \underset{i=1,2,\ldots,T}{\operatorname{argmax}}\ (\mathbf{W}\mathcal{F}(x) + \mathbf{b})_i, \tag{5}$$

where $\mathbf{W} = \mathbf{M}^{(1:T)}\Lambda^{(1:T)}$, rows of $\mathbf{M}^{(1:T)}$ are mean vectors $\mu^{(t)}$ $(t = 1, 2, \ldots, T)$, and $\mathbf{b}_i = -\frac{1}{2}\mu^{(i)}\Lambda^{(1:T)}\mu^{(i)}$.

Unlike previous task prediction strategies (Wortsman et al., 2020; Rajasegaran et al., 2020), with LDA we can predict the task ID with a single test sample, rather than with a batch of samples, representing an important advantage. This is possible because each task consists of defects of the same type and can be described well by a normal distribution with class means $\mu^{(t)}$ and common covariance matrix $\Sigma^{(1:T)}$.

Experiments and results

We evaluate our LDA-CP&S approach on the SD-saliency-900 (Song et al., 2020) and Magnetic tile defects (Huang et al., 2020) datasets, comparing with the following scenarios:

- joint training: the model has access to all data at each incremental step. This case is an upper bound for rehearsal-based methods.
- finetuning: the model is trained at each incremental step t without preventing forgetting, i.e., we finetune the model to a new task t that is pretrained on previous tasks $1, 2, \ldots, t-1$, inevitably causing forgetting of previous tasks because the network parameters (weights and biases) are updated for task t.
- Regularization-based continual learning methods: LwF, MAS that penalize important parameters from changing

(see “Continual learning” Section, Eq. 1), in an attempt to alleviate forgetting.

We do not consider rehearsal-based approaches that replay a small portion of data from previous tasks while learning a new one because our premise is that old data is not available and should not be used. Furthermore, our comparative investigation of the proposed LDA-CP&S method with others includes the joint training strategy, which is an upper bound for rehearsal-based methods, where all data is available at each incremental step. Therefore, if we show that LDA-CP&S performs similarly to joint training, there is no need to consider rehearsal-based continual learning methods.

As performance metrics, we follow other segmentation works and use the mean Pixel accuracy, Dice and Intersection over Union scores. For ground truth $Y = (y_{ij})_{i,j=1}^{H,W}$ and prediction $\hat{Y} = (\hat{y}_{ij})_{i,j=1}^{H,W}$ $(y_{ij}, \hat{y}_{ij} \in \{0, 1\})$, Pixel accuracy, Dice and IoU scores are computed as follows:

$$\text{Pixel accuracy}(\hat{Y}, Y) = 100\% \cdot \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} \mathbf{1}\{y_{ij} == \hat{y}_{ij}\}, \tag{6}$$

$$\text{Dice}(\hat{Y}, Y) = \frac{2|Y \cap \hat{Y}|}{|Y| + |\hat{Y}|}, \tag{7}$$

$$\text{IoU}(\hat{Y}, Y) = \frac{|Y \cap \hat{Y}|}{|Y \cup \hat{Y}|}, \tag{8}$$

where $\mathbf{1}$ is an indicator function and H and W are the height and width of the output image.

To train the model, we use IoU loss which leads to better performance in our experiments than other losses, e.g., Tversky loss (Salehi et al., 2017) and Focal loss (Lin et al., 2017). However, it is worth noting that the difference in IoU scores between models trained with different loss functions is not significant. The IoU loss is computed as follows:

$$\text{IoUloss}(\hat{P}, Y) = 1 - \frac{\sum_{i=1}^{H} \sum_{j=1}^{W} \hat{p}_{ij} \cdot y_{ij} + \varepsilon}{\sum_{i=1}^{H} \sum_{j=1}^{W} \hat{p}_{ij} + y_{ij} - \hat{p}_{ij} \cdot y_{ij} + \varepsilon}, \tag{9}$$

where $\hat{p}_{ij} \in [0, 1]$ are the output probabilities, and H and W are the height and width of the output image, ε is a smoothing parameter.

SD-saliency-900 dataset

In the case of the SD-saliency-900 dataset, we consider a smaller version of U-Net with 16, 32, 64 and 128 in the encoder block and 256 channels in the bottleneck because it consists of only three types of defects – Scratches, Patches and Inclusion – with 300 images per defect. The original size of the images is 200 × 200 but we resize the images to 224 × 224 to make them acceptable for U-Net. We train the segmentation model for 70 epochs with 8 images in a batch, using Adam (Kingma and Ba, 2015) optimizer and learning rate 0.001. During the pruning stage, we use $\alpha = 0.9$ and 3 pruning iterations. More details about the influence of hyperparameters on the results are shown in Section 4.3. As it is common in continual learning literature (Masana et al., 2020), we consider different task orderings in our experiments. We can construct six task orderings for the current dataset (e.g., Patches → Scratches → Inclusion). For other approaches with which we compare our method, the training hyperparameters such as the number of training epochs, learning rate and optimizer are the same as for LDA-CP&S.

Table 1 Classification accuracy (%) for SD-saliency-900 dataset. The numbers are averaged over all six orderings

	Scratches	Patches	Inclusion	Average
accuracy (%)	98.33	100	100	99.44

First, we have to make sure that LDA can accurately predict the defect type in an incremental manner. Table 1 illustrates the classifier’s accuracy for each defect averaged over all six orders. We can observe the high performance of LDA, misclassifying only a few images from the Scratches dataset. Since 60 images were selected to test each defect type, the prediction error presented corresponds to only 1 misclassified image.

In Fig. 4, we show mIoU score after every incremental step for every task order. Regularization-based methods only slightly outperform finetuning strategy, while our LDA-CP&S shows comparable results to joint training. Poor performance of the regularization methods can be explained by the lack of a task-specific output layer, which is present in classification network architectures as a classification head. Therefore MAS and LwF update all parameters but change them slightly less than finetuning. On the contrary, LDA-CP&S creates fixed task-specific subnetworks that can overlap and transfer knowledge between each other. Since LDA predicts the defect type (i.e., subnetwork) well at the inference stage, we almost do not have any losses in segmentation performance. We also do not observe network saturation, i.e., the situation when the model does not have enough free space to learn a new task, even though we use a smaller version of U-Net. Table 2 summarizes the final pixel accuracy, Dice and mIoU average scores for all considered learning cases, including the single-task scenario where we train six separate models (one per defect type). Notably, LDA-CP&S not only outperforms the regularization-based approaches but is also more robust to different task order-

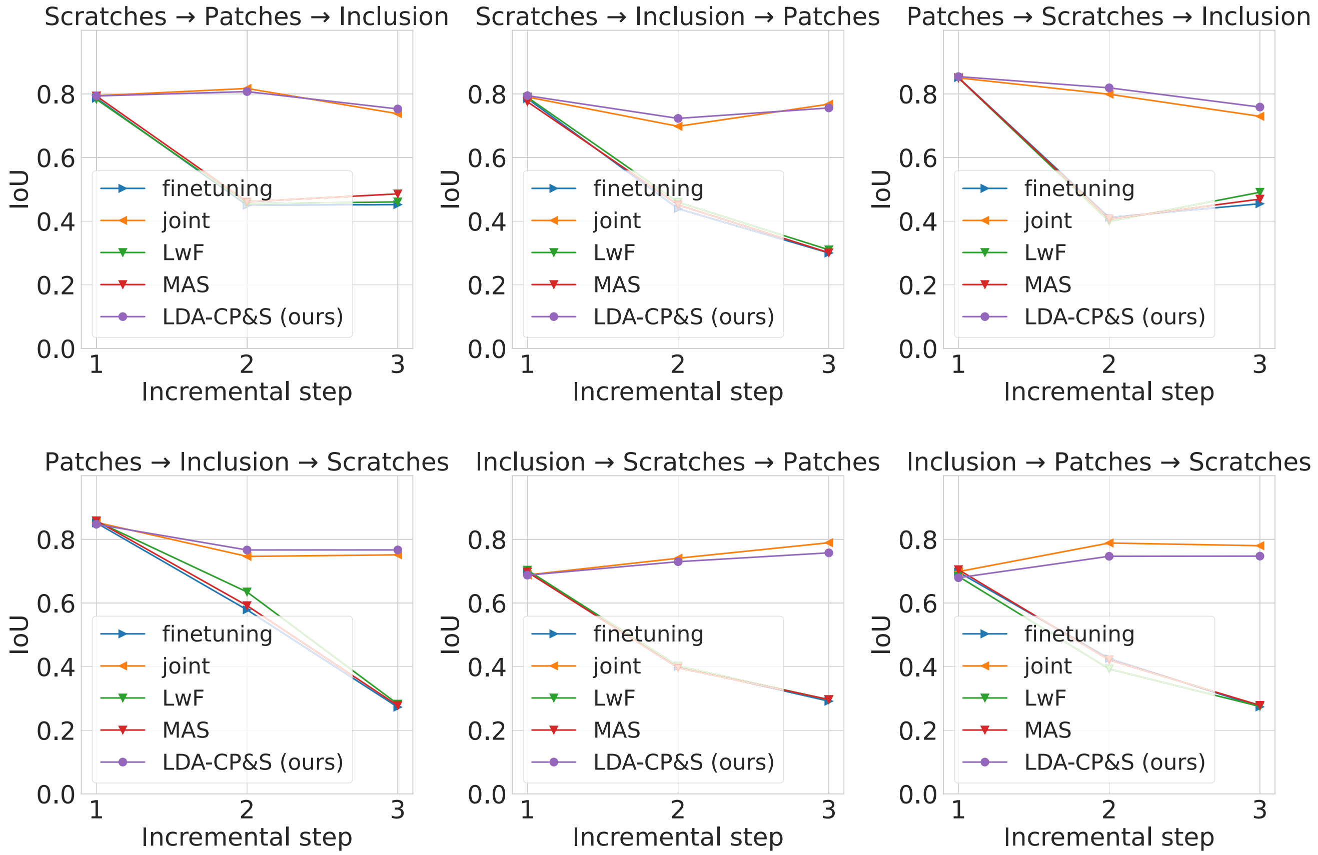

Fig. 4 IoU score after every incremental step for SD-saliency-900 dataset. The results are presented for all six possible defect orderings

Table 2 Pixel accuracy, Dice and IoU scores averaged over 6 orderings (± standard deviation) after all six tasks are learned based on the SD-saliency-900 dataset

	Pixel accuracy	Dice	mIoU
single-task	97.54 ± 0.92	86.44 ± 5.41	77.92 ± 7.17
joint	97.33 ± 0.28	84.94 ± 1.97	75.93 ± 2.38
LwF (Li and Hoiem, 2017)	85.81 ± 3.36	42.70 ± 13.62	35.25 ± 9.67
MAS (Aljundi et al., 2018)	84.86 ± 7.17	42.35 ± 14.24	35.15 ± 9.83
LDA-CP&S (ours)	**97.19** ± 0.16	**84.77** ± 0.40	**75.65** ± 0.65

ing, having significantly lower standard deviation across all learning scenarios.

Magnetic tile defects dataset

Magnetic tile defects dataset (Huang et al., 2020) contains five types of defects, namely Blowhole, Break, Crack, Fray and Uneven, and images that are free from defects (Free). In this work, we consider only images with defects, i.e., five classes. Since the number of defects is higher in this case, we use a U-Net of the original size with 64, 128, 256, and 512 in the encoder block and 1024 channels in the bottleneck. All images in the dataset have different image sizes and, therefore, we resize them to 224 × 224. For every defect, we randomly select 80% images for training and the rest for testing. U-Net is trained for 150 epochs with 8 images in a batch, using Adam optimizer and learning rate 0.0001. We use these hyperparameters for all considered training methods. Since the total number of possible task orderings is quite large ($5! = 120$), we consider only five of them at random and we do not have reason to believe that the final performance would be very different when choosing other orderings:

- Blowhole → Break → Crack → Fray → Uneven;
- Break → Uneven → Fray → Crack → Blowhole;
- Crack → Blowhole → Break → Uneven → Fray;
- Fray → Crack → Uneven → Blowhole → Break;
- Uneven → Fray → Blowhole → Break → Crack,

Table 3 Classification accuracy (%) for Magnetic tile dataset and the total size of the dataset. The numbers for accuracy are averaged over all five orderings

	Blowhole	Break	Crack	Fray	Uneven	Average
# train (test) images	92 (23)	92 (22)	68 (17)	25 (7)	72 (21)	N/A
test accuracy (%)	100	100	100	85.71	100	98.75

Table 4 Pixel accuracy, Dice and IoU scores averaged over 5 orderings ± standard deviation after all five tasks are learned based on Magnetic tile defects dataset

	Pixel accuracy	Dice	mIoU
single-task	98.70 ± 1.68	88.96 ± 5.93	83.47 ± 6.11
joint	98.52 ± 0.56	86.25 ± 1.58	79.63 ± 1.51
LwF (Li and Hoiem, 2017)	90.49 ± 1.71	36.43 ± 9.58	30.43 ± 8.55
MAS (Aljundi et al., 2018)	91.22 ± 1.19	37.81 ± 7.20	31.56 ± 6.57
LDA-CP&S (ours)	**98.25** ± 0.19	**87.22** ± 1.12	**80.91** ± 1.68

where each defect type appears exactly once for each ordering.

One of the main difficulties with this dataset is class imbalance. Table 3 presents LDA accuracy with the same feature extractor considered in the previous example: the pretrained EfficientNet-B5 architecture. Overall, our classification model is able to identify correctly four out of five types of defects, having some difficulties with the Fray sub-dataset that contains the smallest number of images. The only mistake was done in the Fray sub-dataset where we have only 7 test images, meaning that only one image is classified wrongly.

We would like to highlight the necessity of pretraining the network that extracts features for LDA. The pretrained EfficientNet-B5 produces lower dimensional embeddings that can be used for training and classification with an accuracy of 98.75%, misclassifying only one test image. Meanwhile, if we were to consider a feature extractor with random parameters it would compress the input images in such a way that the LDA classifier would only achieve 16.24% of accuracy.

Table 4 illustrates the final average scores after the model learned all five tasks. As in the previous case, LDA-CP&S significantly outperforms the regularization-based methods and shows better robustness to different task ordering. In Fig. 5, we present the mIoU score after every incremental step, comparing our LDA-CP&S with other continual learning methods. As we saw in the previous example, regularization-based methods do not handle this type of segmentation problem well. The tasks that we constructed from the Magnetic tile dataset can be quite dissimilar having significant differences in defect areas. Therefore, by updating all the parameters without having task-specific ones, regularization-based approaches are only slightly better than simple finetuning where no anti-forgetting measures are considered. In contrast, our LDA-CP&S creates task-specific parameters for each defect, fixing the values of the parameters once they are assigned to a subnetwork (i.e., defect type or task ID). This allows LDA-CP&S to deal with sequences of tasks as well as joint training, which is very encouraging because joint training is a performance upper bound since all the data is available at each incremental step.

We also investigated how the mIoU score changes for every task after each incremental step. In Fig. 6, we consider one of the task orderings: Fray → Crack → Uneven → Blowhole → Break. The figure clearly shows the advantage of our algorithm over regularization-based ones because they are heavily dependent on the similarity of the tasks in the order. For example, learning the Break sub-dataset (the last incremental step) improves performance on Fray and Crack sub-datasets compared to the previous incremental step for MAS, LwF and finetuning strategies. However, Uneven is totally forgotten after the network is trained on the Blowhole sub-dataset.

On the contrary, LDA-CP&S does not forget previous tasks and is still able to learn new ones even having fewer free parameters. It has comparable performance with a single-task scenario, where a separate U-Net is trained for every task. Also, we observe that task-wise performance is almost the same as for joint training, meaning the subnetwork overlaps provide enough knowledge transfer to learn a new task. As in the previous case, we did not experience the network saturation issue, meaning that the current architecture has not reached the limit yet. However, if the process were to continue, after a certain number of tasks the model would not learn new defects effectively. This was observed with the CP&S method (Dekhovich et al., 2023) applied to the CIFAR-100 classification problem (Krizhevsky, 2009) when the dataset was split into 20 tasks.

Figure 7 illustrates the model output in every learning scenario. We observe that regularization-based methods and finetuning cannot capture the defects of the first tasks in the sequence, while our LDA-CP&S finds defects' segments close to the joint training.

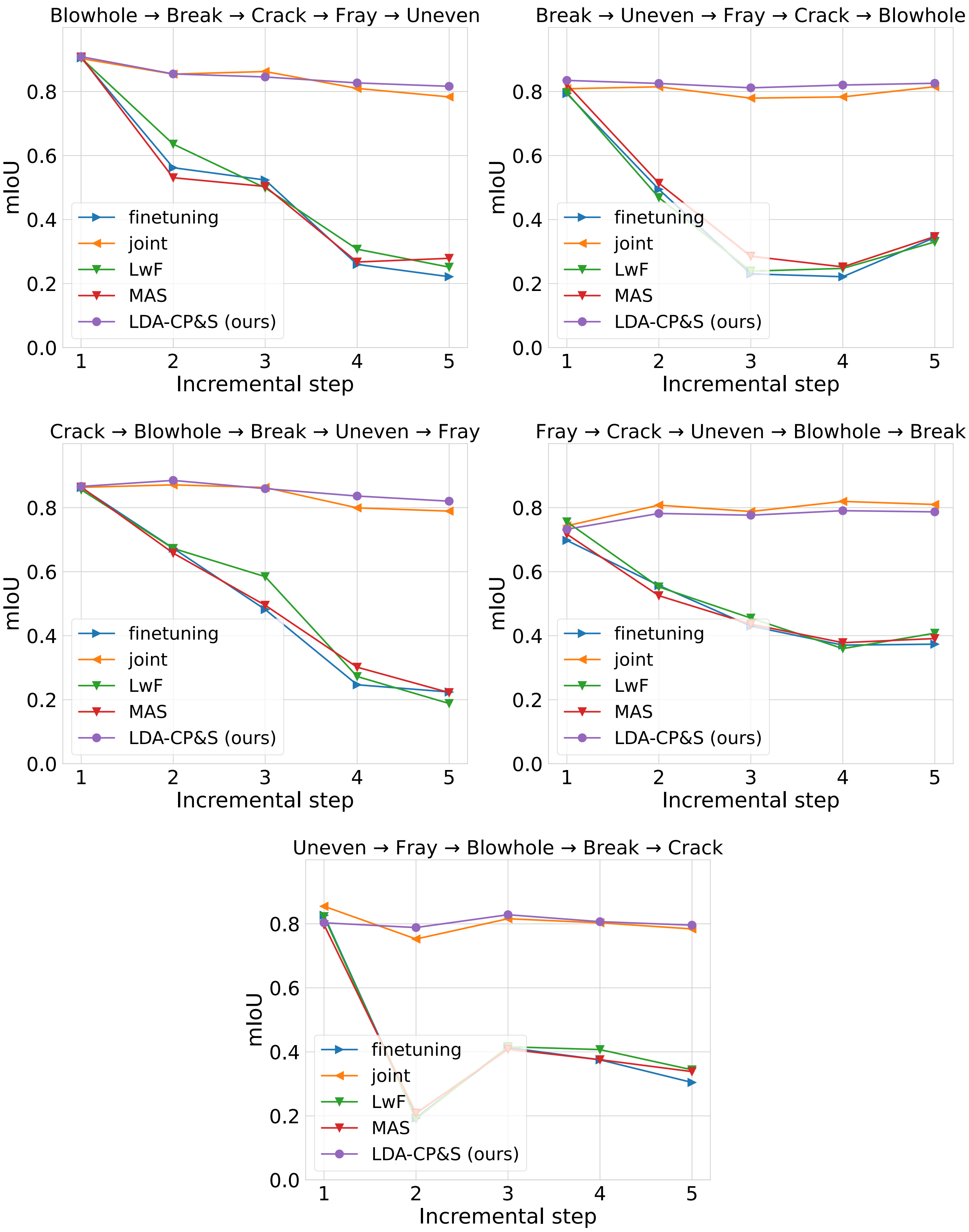

Fig. 5 IoU score after every incremental step for Magnetic tile datasets. The results are shown for all five selected defect orderings

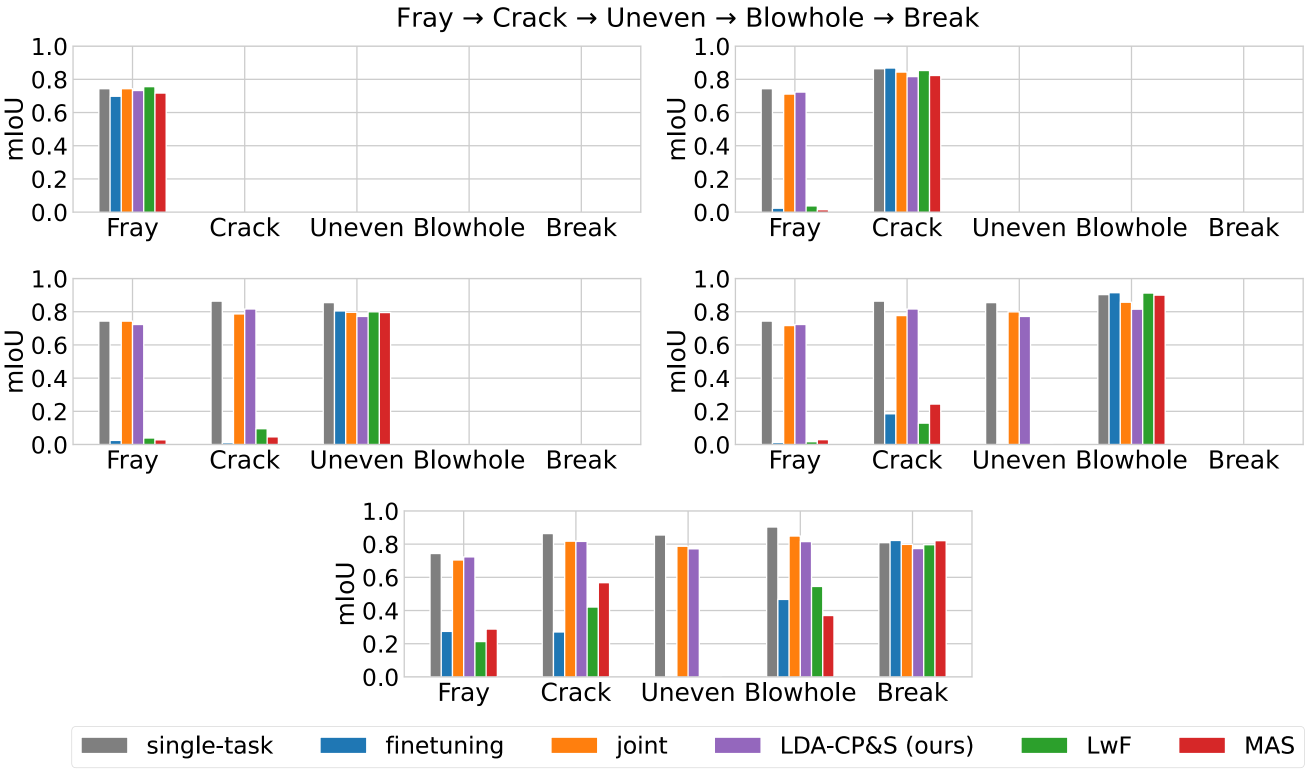

Fig. 6 IoU score after every incremental step for one of the defects orderings from the Magnetic tile dataset

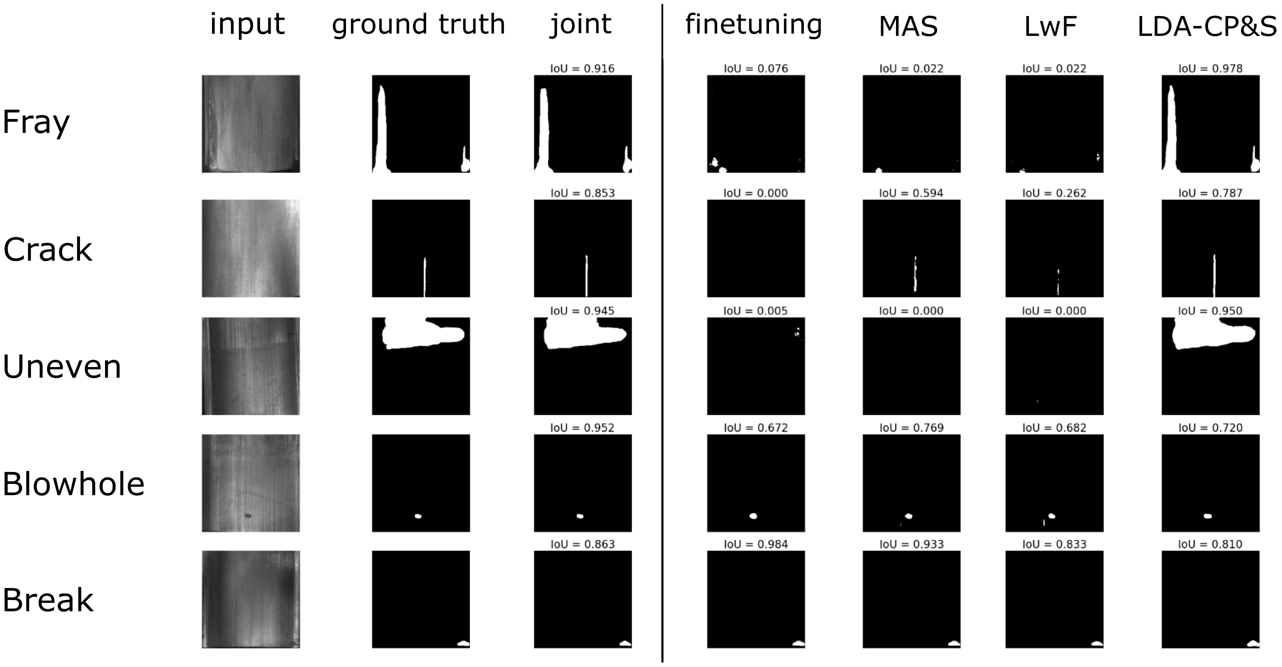

Fig. 7 Visualization of defects prediction for the considered scenarios on Fray → Crack → Uneven → Blowhole → Break task ordering

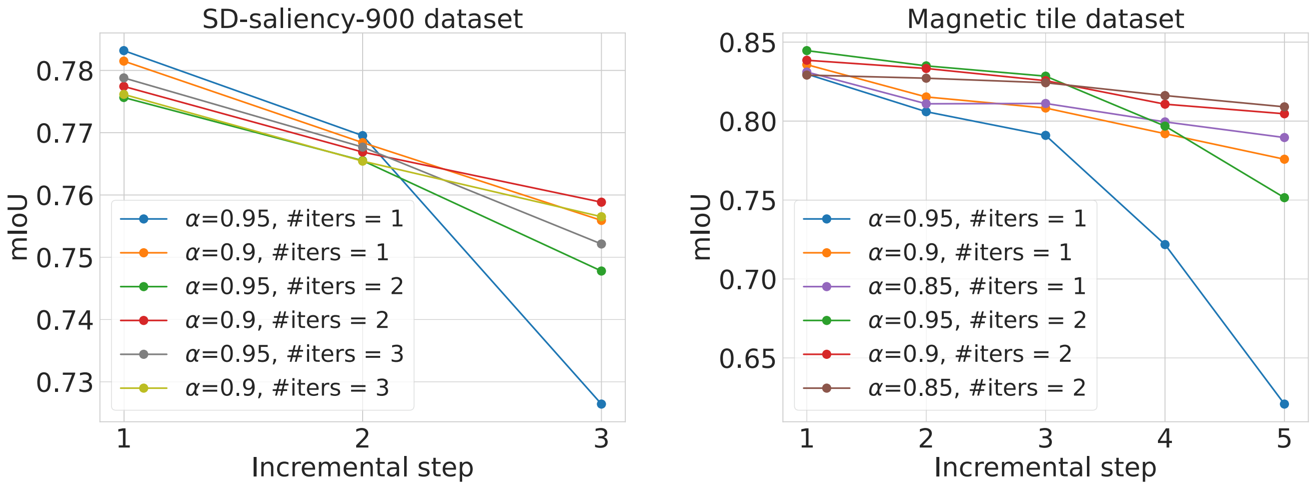

Fig. 8 Comparison between different pairs of hyperparameters for the pruning step in our LDA-CP&S method on SD-saliency-900 dataset (left) and Magnetic tile dataset (right). The same colors on both figures correspond to the same hyperparameter values; note that the same hyperparameter choice is near optimal on both datasets (red line). The results after each incremental step averaged over the number of considered task orderings

Hyperparameters choice

The choice of hyperparameters for pruning has a significant impact on subnetwork sparsity and, as a result, performance. In this subsection, we compare different options for the pruning hyperparameter α and the number of pruning iterations. A lower number of α and a higher number of pruning iterations lead to higher sparsity (more free connections to learn future tasks) but may cause lower segmentation performance. Also, the values for hyperparameters depend on the length of task sequences. In our work, we pre-define these hyperparameters at the beginning and do not change them during the training process.

Figure 8 illustrates how different pairs of hyperparameters affect the training process for our approach. For both datasets, we clearly see that the network starts to saturate if pruning is not aggressive enough (e.g., $\alpha = 0.95$ where most of the signal is conserved) because the network does not have enough free parameters for new tasks. In the case of the SD-saliency-900 dataset, we can also observe the trade-off between sparsity and mIoU score: with $\alpha = 0.9$ it is clear that pruning the network twice leads to better performance than doing it three times, as the subnetwork that results is less expressive (has fewer parameters). The results on the Magnetic tile dataset show the trade-off between learning the first tasks and the last ones: if we prune the network twice, $\alpha = 0.9$ leads to better performance if there are no more than three tasks, while $\alpha = 0.85$ is better suitable for longer task sequences.

Conclusion

We believe smart monitoring systems should quickly adapt to new tasks without a dramatic drop in performance on previously learned ones. However, this is not the case based on the current state of the literature on surface defect inspection. Thus, there is a need for continual learning of deep neural networks for automatic surface defect segmentation such that product quality assessment is improved. By training deep learning models incrementally, we show that we can accumulate all the learned information without retraining when a new task comes to the network. In addition, we do not need to store data for retraining, which can be either not allowed or not possible due to the (lack of) availability of old datasets.

The LDA-CP&S method that we propose successfully learns to segment the defects incrementally, without any forgetting, using only the data that is given at the current time step. Meanwhile, other methods that do not use data from previous tasks fail to remember all tasks, exhibiting considerable forgetting in segmenting previously seen defects. Overall, the performance of LDA-CP&S is more than two times higher in terms of mean Intersection over Union score for the two datasets considered herein when compared to other continual learning methods. Moreover, it is comparable with joint training where the model has access to all the data observed up to the current incremental step.

Appendix A Abbreviations and Variables

The table 5 shows the abbreviations and variables with their meanings that are most often used in the article.

Table 5 Abbreviations and Variables

Abbreviation	Meaning
CP&S	Continual Prune-and-Select Dekhovich et al. (2023)
IoU	Intersection-over-Union (metrics)
LDA	Linear Discriminant Analysis
LwF	Learning without Forgetting Li and Hoiem (2017)
MAS	Memory Aware Synopsis Aljundi et al. (2018)
Variable	Meaning
α	pruning parameter
λ	regularization parameter for MAS and LwF
num_iters	number of pruning iterations

Author Contributions A.D. performed the research and wrote the paper, A.D. and M.A.B. designed the research, M.A.B. supervised the work and edited the paper.

Funding Not applicable

Availability of data and materials The information about data used for this research is provided in the manuscript.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethics approval Not applicable

Consent to participate Not applicable

Consent for publication Not applicable

Code availability Code implementation is available at:https://github.com/adekhovich/continual_defect_segmentation.

References

Agarwal, K., Shivpuri, R., Zhu, Y., Chang, T.-S., & Huang, H. (2011). Process knowledge based multi-class support vector classification (pk-msvm) approach for surface defects in hot rolling. *Expert Systems with Applications, 38*(6), 7251–7262. https://doi.org/10.1016/j.eswa.2010.12.026

Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., & Tuytelaars, T. (2018). Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 139–154. https://doi.org/10.1007/978-3-030-01219-9_9

Aslam, M., Khan, T. M., Naqvi, S. S., Holmes, G., & Naffa, R. (2020). Ensemble convolutional neural networks with knowledge transfer for leather defect classification in industrial settings. *IEEE Access, 8*, 198600–198614. https://doi.org/10.1109/ACCESS.2020.3034731

Baweja, C., Glocker, B., & Kamnitsas, K. (2018). Towards continual learning in medical imaging. In *Medical Imaging Meets NIPS Workshop*, 32nd Conference on Neural Information Processing Systems (NIPS)

Castro, F.M., Marín-Jiménez, M.J., Guil, N., Schmid, C., & Alahari, K. (2018). End-to-end incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 233–248. https://doi.org/10.1007/978-3-030-01258-8_15

Chao, S.-M., & Tsai, D.-M. (2008). An anisotropic diffusion-based defect detection for low-contrast glass substrates. *Image and Vision Computing, 26*(2), 187–200. https://doi.org/10.1016/j.imavis.2007.03.003

Cha, S., Yoo, Y., & Moon, T. (2021). Ssul: Semantic segmentation with unknown label for exemplar-based class-incremental learning. *Advances in neural information processing systems, 34*, 10919–10930.

Coop, R., Mishtal, A., & Arel, I. (2013). Ensemble learning in fixed expansion layer networks for mitigating catastrophic forgetting. *IEEE transactions on neural networks and learning systems, 24*(10), 1623–1634. https://doi.org/10.1109/TNNLS.2013.2264952

Dasgupta, S., & Hsu, D. (2007). On-line estimation with the multivariate gaussian distribution. In *International Conference on Computational Learning Theory*, pp. 278–292. https://doi.org/10.1007/978-3-540-72927-3_21 . Springer

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., & Tuytelaars, T. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence, 44*(7), 3366–3385. https://doi.org/10.1109/TPAMI.2021.3057446

Dekhovich, A., Tax, D. M., Sluiter, M. H., & Bessa, M. A. (2023). Continual prune-and-select: class-incremental learning with specialized subnetworks. *Applied Intelligence, 53*(14), 17849–17864. https://doi.org/10.1007/s10489-022-04441-z

Dekhovich, A., Tax, D. M., Sluiter, M. H., & Bessa, M. A. (2024). Neural network relief: a pruning algorithm based on neural activity. *Machine Learning*. https://doi.org/10.1007/s10994-024-06516-z

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. https://doi.org/10.1109/CVPR.2009.5206848 . Ieee

Dorfer, M., Kelz, R., & Widmer, G. (2016). Deep linear discriminant analysis. In *4th International Conference on Learning Representations*, ICLR

Douillard, A., Chen, Y., Dapogny, A., & Cord, M. (2021). Plop: Learning without forgetting for continual semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4040–4050

Douillard, A., Cord, M., Ollion, C., Robert, T., & Valle, E. (2020). Podnet: Pooled outputs distillation for small-tasks incremental learning. In Computer Vision-ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, (2020). *Proceedings, Part XX,16*, 86–102. https://doi.org/10.1007/978-3-030-58565-5_6.Springer

Feng, X., Gao, X., & Luo, L. (2021). X-sdd: A new benchmark for hot rolled steel strip surface defects detection. *Symmetry, 13*(4), 706. https://doi.org/10.3390/sym13040706

French, R. M. (1999). Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences, 3*(4), 128–135. https://doi.org/10.1016/S1364-6613(99)01294-2

Garderen, K., Voort, S., Incekara, F., Smits, M., & Klein, S. (2019). Towards continuous learning for glioma segmentation with elastic weight consolidation. In *International Conference on Medical Imaging with Deep Learning -Extended Abstract Track*

Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., & Bengio, Y. (2014). An empirical investigation of catastrophic forgetting in gradient-based neural networks. In *2nd International Conference on Learning Representations*, ICLR

Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. *Advances in neural information processing systems***28**

Hao, R., Lu, B., Cheng, Y., Li, X., & Huang, B. (2021). A steel surface defect inspection approach towards smart industrial monitoring. *Journal of Intelligent Manufacturing, 32*, 1833–1843. https://doi.org/10.1007/s10845-020-01670-2

Hayes, T.L., & Kanan, C. (2020). Lifelong machine learning with deep streaming linear discriminant analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 220–221

He, Y., Song, K., Meng, Q., & Yan, Y. (2019). An end-to-end steel surface defect detection approach via fusing multiple hierarchical features. *IEEE transactions on instrumentation and measurement, 69*(4), 1493–1504. https://doi.org/10.1109/TIM.2019.2915404

He, D., Xu, K., & Zhou, P. (2019). Defect detection of hot rolled steels with a new object detection framework called classification priority network. *Computers & Industrial Engineering, 128*, 290–297. https://doi.org/10.1016/j.cie.2018.12.043

Huang, Y., Qiu, C., & Yuan, K. (2020). Surface defect saliency of magnetic tile. *The Visual Computer, 36*, 85–96. https://doi.org/10.1007/s00371-018-1588-5

Jeon, Y.-J., Choi, D.-C., Lee, S. J., Yun, J. P., & Kim, S. W. (2014). Defect detection for corner cracks in steel billets using a wavelet reconstruction method. *JOSA A, 31*(2), 227–237. https://doi.org/10.1364/JOSAA.31.000227

Jia, H., Murphey, Y.L., Shi, J., &Chang, T.-S. (2004). An intelligent real-time vision system for surface defect detection. In *Proceedings of the 17th International Conference on Pattern Recognition*, 2004. ICPR 2004., vol. 3, pp. 239–242. https://doi.org/10.1109/ICPR.2004.1334512 . IEEE

Kim, E.S., Kim, J.U., Lee, S., Moon, S.-K., & Ro, Y.M. (2020). Class incremental learning with task-selection. In *2020 IEEE International Conference on Image Processing (ICIP)*, pp. 1846–1850. https://doi.org/10.1109/ICIP40778.2020.9190703 . IEEE

Kim, G., Xiao, C., Konishi, T., Ke, Z., & Liu, B. (2022). A theoretical study on solving continual learning. *Advances in Neural Information Processing Systems, 35*, 5065–5079.

Kingma, D.P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, ICLR

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., & Grabska-Barwinska, A. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences, 114*(13), 3521–3526. https://doi.org/10.1073/pnas.1611835114

Klingner, M., Bär, A., Donn, P., & Fingscheidt, T. (2020). Class-incremental learning for semantic segmentation re-using neither old data nor old labels. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems* (ITSC), pp. 1–8 . https://doi.org/10.1109/ITSC45102.2020.9294483 . IEEE

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Master's thesis, University of Toronto

Lee, N., Ajanthan, T., & Torr, P.H. (2019). Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations* (ICLR)

Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H.P. (2017). Pruning filters for efficient convnets. In *5th International Conference on Learning Representations*, ICLR

Li, Z., & Hoiem, D. (2017). Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence, 40*(12), 2935–2947. https://doi.org/10.1109/TPAMI.2017.2773081

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988. https://doi.org/10.1109/ICCV.2017.324

Liu, T., & Ye, W. (2022). A semi-supervised learning method for surface defect classification of magnetic tiles. *Machine Vision and Applications, 33*(2), 35. https://doi.org/10.1007/s00138-022-01286-x

Lv, X., Duan, F., Jiang, J.-J., Fu, X., & Gan, L. (2020). Deep metallic surface defect detection: The new benchmark and detection network. *Sensors, 20*(6), 1562. https://doi.org/10.3390/s20061562

Mallya, A., & azebnik, S. (2018). Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7765–7773

Masana, M., Twardowski, B., & Weijer, J. (2020). On class orderings for incremental learning. arXiv preprint arXiv:2007.02145

Maschler, B., Pham, T. T. H., & Weyrich, M. (2021). Regularization-based continual learning for anomaly detection in discrete manufacturing. *Procedia CIRP, 104*, 452–457. https://doi.org/10.1016/j.procir.2021.11.076

Maschler, B., Tatiyosyan, S., & Weyrich, M. (2022). Regularization-based continual learning for fault prediction in lithium-ion batteries. *Procedia CIRP, 112*, 513–518. https://doi.org/10.1016/j.procir.2022.09.091

Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence, 24*(7), 971–987. https://doi.org/10.1109/TPAMI.2002.1017623

Pan, Y., & Zhang, L. (2022). Dual attention deep learning network for automatic steel surface defect segmentation. *Computer-Aided Civil and Infrastructure Engineering, 37*(11), 1468–1487. https://doi.org/10.1111/mice.12792

Prunella, M., Scardigno, R. M., Buongiorno, D., Brunetti, A., Longo, N., Carli, R., Dotoli, M., & Bevilacqua, V. (2023). Deep learning for automatic vision-based recognition of industrial surface defects: a survey. *IEEE Access*. https://doi.org/10.1109/ACCESS.2023.3271748

Qiu, Y., Shen, Y., Sun, Z., Zheng, Y., Chang, X., Zheng, W., & Wang, R. (2023). Sats: Self-attention transfer for continual semantic segmentation. *Pattern Recognition, 138*, 109383. https://doi.org/10.1016/j.patcog.2023.109383

Rajasegaran, J., Khan, S., Hayat, M., Khan, F.S., & Shah, M. (2020). itaml: An incremental task-agnostic meta-learning approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13588–13597

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., & Lampert, C.H. (2017). icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2001–2010. https://doi.org/10.1109/CVPR.2017.587

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention-MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015. *Proceedings, Part III,18*, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28.Springer

Salehi, S.S.M., Erdogmus, D., & Gholipour, A. (2017). Tversky loss function for image segmentation using 3d fully convolutional deep networks. In *International Workshop on Machine Learning in Medical Imaging*, pp. 379–387. https://doi.org/10.1007/978-3-319-67389-9_44 . Springer

Shanmugamani, R., Sadique, M., & Ramamoorthy, B. (2015). Detection and classification of surface defects of gun barrels using computer vision and machine learning. *Measurement, 60*, 222–230. https://doi.org/10.1016/j.measurement.2014.10.009

Sokar, G., Mocanu, D.C., & Pechenizkiy, M. (2022). Avoiding forgetting and allowing forward transfer in continual learning via sparse networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 85–101. https://doi.org/10.1007/978-3-031-26409-2_6 . Springer

Song, G., Song, K., & Yan, Y. (2020). Edrnet: Encoder-decoder residual network for salient object detection of strip steel surface defects. *IEEE Transactions on Instrumentation and Measurement, 69*(12), 9709–9719. https://doi.org/10.1109/TIM.2020.3002277

Song, G., Song, K., & Yan, Y. (2020). Saliency detection for strip steel surface defects using multiple constraints and improved texture features. *Optics and Lasers in Engineering, 128*, 106000. https://doi.org/10.1016/j.optlaseng.2019.106000

Song, K., & Yan, Y. (2013). A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. *Applied Surface Science, 285*, 858–864. https://doi.org/10.1016/j.apsusc.2013.09.002

Sun, W., Al Kontar, R., Jin, J., & Chang, T.-S. (2023). A continual learning framework for adaptive defect classification and inspection. *Journal of Quality Technology, 55*(5), 598–614. https://doi.org/10.1080/00224065.2023.2224974

Tabernik, D., Šela, S., Skvarč, J., & Skočaj, D. (2020). Segmentation-based deep-learning approach for surface-defect detection. *Journal of Intelligent Manufacturing, 31*(3), 759–776. https://doi.org/10.1007/s10845-019-01476-x

Tan, M., & Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6105–6114. PMLR

Tercan, H., Deibert, P., & Meisen, T. (2022). Continual learning of neural networks for quality prediction in production using memory aware synapses and weight transfer. *Journal of Intelligent Manufacturing, 33*(1), 283–292. https://doi.org/10.1007/s10845-021-01793-0

Thrun, S., & Pratt, L. (1998). (eds.): Learning to Learn. Springer US, Boston, MA. https://doi.org/10.1007/978-1-4615-5529-2

Üzen, H., Türkoğlu, M., Yanikoglu, B., & Hanbay, D. (2022). Swinmfinet: Swin transformer based multi-feature integration network for detection of pixel-level surface defects. *Expert Systems with Applications, 209*, 118269. https://doi.org/10.1016/j.eswa.2022.118269

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems,30*

Wang, F.-Y., Zhou, D.-W., Ye, H.-J., & Zhan, D.-C. (2022). Foster: Feature boosting and compression for class-incremental learning. In *European Conference on Computer Vision*, pp. 398–414. https://doi.org/10.1007/978-3-031-19806-9_23 . Springer

Wortsman, M., Ramanujan, V., Liu, R., Kembhavi, A., Rastegari, M., Yosinski, J., & Farhadi, A. (2020). Supermasks in superposition. *Advances in Neural Information Processing Systems, 33*, 15173–15184.

Wu, H., & Lv, Q. (2021). Hot-rolled steel strip surface inspection based on transfer learning model. *Journal of Sensors, 2021*, 1–8.

Yan, S., Xie, J., & He, X. (2021) Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3014–3023

Yan, S., Zhou, J., Xie, J., Zhang, S., & He, X. (2021). An em framework for online incremental learning of semantic segmentation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 3052–3060. https://doi.org/10.1145/3474085.3475443

Yoon, J., Yang, E., Lee, J., & Hwang, S.J. (2018). Lifelong learning with dynamically expandable networks. In 6th International Conference on Learning Representations, ICLR

Zenke, F., Poole, B., &Ganguli, S. (2017). Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR

Zhang, J., Zhang, J., Ghosh, S., Li, D., Tasci, S., Heck, L., Zhang, H., & Kuo, C.-C.J. (2020). Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1131–1140

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.