Machine Learning (2024) 113:4013–4035
https://doi.org/10.1007/s10994-023-06498-4

On-the-fly image-level oversampling for imbalanced datasets of manufacturing defects

Spyros Theodoropoulos[1,2] · Patrik Zajec[3] · Jože M. Rožanec[3] · Dimosthenis Kyriazis[2] · Panayiotis Tsanakas[1]

Received: 8 June 2022 / Revised: 22 September 2023 / Accepted: 16 December 2023 / Published online: 17 January 2024

Abstract
Visual defect recognition and its manufacturing applications have been an upcoming topic in recent AI research. Defect datasets are often severely imbalanced and can be additionally burdened with separating classes of high visual similarity. Although various methods of data augmentation have been proposed to mitigate the class imbalance, they often fail to cope with tinier minority classes or have fidelity issues with smaller defects while, at the same time, needing significant computational resources to train. Also, augmentation based on vector-based oversampling struggles to produce high-fidelity inputs and is hard to apply on custom CNN architectures, which often perform better for this type of problem. Our work presents an image-level oversampling method based on an instance-based image generator that can be applied to any CNN directly during the training process without increasing the order of training time required. It is based on identifying a small number of the most uncertain base samples close to the estimated class boundaries and using them as seeds for augmentation. The resulting images are of high visual quality preserving small class differences, and they also improve the classifier boundary leading to higher recall scores than other state-of-the-art approaches.

Keywords Imbalanced learning · Defect recognition · Data augmentation · Oversampling · Deep learning · Generative adversarial networks

1 Introduction

Automatically detecting and classifying object defects is an important application of modern manufacturing AI systems that presents unique challenges, such as severe class imbalance, high inter-class similarity, and a requirement for high classification performance in real-life settings. Addressing these challenges can provide novel insights and improvements in the general context of imbalanced learning. Class imbalance is an inherent and very frequent issue in datasets of defects used for automated visual quality inspection owing to the rarity of defect occurrences in real-life processes (Fathy et al., 2021). For instance, in many

Editors: Nuno Moniz, Paula Branco, Luís Torgo, Nathalie Japkowicz, Michal Wozniak, Shuo Wang.

Extended author information available on the last page of the article modern manufacturing processes, a defect may occur in one per thousand manufactured objects making the collection of sufficient data for a balanced dataset either too costly or in the worst case nearly impossible. Even though defects are rare, the ability to detect them automatically or in a synergistic way between human and AI algorithms is of great value, since, it not only reduces costs and worker fatigue but also frees up human resources to perform more challenging, less repetitive, and more creative work (See, 2012).

Early approaches in automated visual inspection did not run into the problem of class imbalance as they mainly relied on traditional computer vision methods using pre-extracted features (Tulbure et al., 2022). These methods were custom-designed using rules derived from an expert's domain knowledge and were completely unsupervised, both regarding feature extraction and rule-based decision-making, with no requirement for collecting training data. Even later, more flexible methods such as Histogram of Gradients (Dalal & Triggs, 2005) and (Viola & Jones, 2001) relied on the extraction of custom features tailored to the problem at hand. However, since the introduction of Deep Convolutional Neural Networks (DCNNs) (Krizhevsky et al., 2017) it was made possible to achieve good accuracy scores by deriving extracted features directly from the training data. Despite the new approach requiring the collection of large amounts of data, in some cases even 10^3 to 10^4 samples, and being very sensitive to class imbalance (Kadar & Onita, 2019), it offers several distinct advantages that have made it very popular in the current research:

1. There is little need for expert domain knowledge during feature extraction or decision-making as DCNNs learn mainly from the data. This avoids the development of complex and error-prone data pipelines.
2. Due to this independence from domain expertise DCNNs can be more easily adapted to tackle similar problems (e.g., defect inspection of a similar but different product produced by the same organization), and can also easily accommodate new defect types, given enough training data, without change to the recognition algorithm.
3. DCNNs can easily adapt to differences in simple visual conditions such as translation and scale (Goodfellow et al., 2016).
4. Knowledge extracted from large datasets can be adapted to smaller datasets through transfer learning, thus, coping to a certain extent with high data requirements.

In visual quality inspection, which is the focus of our work, the most frequent approach in the current literature, aimed at mitigating class imbalance, is data augmentation (Zhang et al., 2021; Saiz et al., 2021). Traditionally, image defect datasets are augmented via various graphical transformations, such as scaling, rotation, translation, shearing, blur, illumination, etc. However, those image-level transformations do not contribute sufficiently to the clearer separation between different classes, especially when the separation depends on higher-level features (Pawara et al., 2017). To overcome the limitations of traditional image processing methods, Convolutional Variational Autoencoders (CVAEs) (Sohn et al., 2015) have been proposed and used successfully in a dataset of metal surfaces (Yun et al., 2020). Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) is another important tool, which can efficiently address different kinds of imbalances such as inter-class, intra-class (e.g., person reidentification), and object and pixel level imbalances for segmentation tasks (Sampath et al., 2021). A third family of methods is based on Neural Style Transfer attempting to fuse a "style" image (defect) and a "content" image. Defects can be generated through global (Liu et al., 2019) and local (Luan et al., 2018) style transfer, using extracted defect patches and suitably placing them on the target object. However many of the above methods still require a significant amount of data (being Deep Learning methods) and may not be suitable for all datasets depending on their degree of imbalance as well as the similarity between classes that makes the generation of high-fidelity images difficult. Such methods are also usually computationally intensive requiring long training times. Nevertheless, many modern GAN architectures can be controlled through manipulation of their latent space and therefore can be suitably adapted to specific problems and potentially also made to work with smaller datasets as described in more detail in the Related Work section.

In our work, we applied data augmentation to mitigate class imbalance in a dataset of logo print images on top of manufactured shaver shells. Following our early experiments we noticed that custom shallow CNN architectures that are trained end-to-end on the dataset at hand achieved the most promising performance, therefore we introduced a data augmentation method compatible with end-to-end training. Our approach's novelty lies in using a small sample GAN introduced in Noguchi and Harada (2019) in a confidence-aware manner. This leads the generator to produce synthetic images based on highly uncertain training samples that lie near the classification boundary. The resulting method achieved promising results against recent and established methods based on deep data generation or vector-based oversampling, while also retaining good computational performance by generating synthetic images on the fly.

2 Related work

The current work builds upon two areas of research. The first is on using GANs for generating defects. GANs have proven very reliable in producing high-quality images and many works have managed to apply them to imbalanced and smaller datasets. We also build upon advances in assessing the reliability of neural network predictions. This line of research focuses on ways to obtain confidence estimates of the network's predictions, which we aim to utilize to bias our generation process towards low-confidence samples.

2.1 GANs in defect generation

GANs have been successfully used in many different industrial, biomedical, and other scenarios to tackle the class imbalance found in defect detection problems. The most straightforward way to use them is by training them on the same set of data as the final detector/classifier and then generating data to augment the initial dataset. A step further is to introduce customizations to control a GAN's output either through manipulation of its latent space or the influence of its loss function. A common example of the latter that is very popular in defect detection is encoder/decoder-based architectures.

2.1.1 Direct data augmentation

A variety of architectures have been tried for direct augmentation, for instance, TransGAN (Jiang et al., 2021), a transformer-based GAN was used in an agricultural setting for detecting fruit surface defects (Wang & Xiao, 2021), as well as CGAN, a class-conditioned architecture, able to more precisely synthesize classes of defects (Bird et al., 2022). A very popular architecture for these scenarios is Deep Convolutional GAN (DCGAN). In a comparative study of steel strip defect detection (Jain et al., 2020), it outperformed models such as the information-theoretic InfoGAN (Chen et al., 2016), and has improved accuracy metrics in imbalanced datasets from a variety of domains such as fiber layup inspection (Meister et al., 2021), liver lesion classification (Frid-Adar et al., 2018) and defect generation (Wang et al., 2021), often trained with the help of additional data augmentation via geometric or stylistic transformations. An improved version of DCGAN, capable of producing more diverse data, Wasserstein GAN (WGAN), was applied to the detection of weld (Zhang et al., 2019) and decorative sheet (Le et al., 2020) defects, however, complicated defects such as "burn-through" and "crack" welding defects still needed to be synthesized graphically using human prior knowledge. Finally, a more recent and sophisticated architecture, StyleGAN2-ADA (Karras et al., 2020), was capable of high-fidelity generation of structural adhesive defects trained over a small input dataset of fewer than two hundred images (Peres et al., 2021), with limited additional augmentation and manual labeling.

2.1.2 Customized architectures

Apart from the direct application of GAN architectures, several customized architectures have been developed to specifically tackle defect synthesis. For instance, AC-PG GAN is a combination of the Progressive Growing GAN (PGGAN) (Karras et al., 2018) and Auxilliary Classifier GAN (ACGAN) (Han et al., 2019) aimed at the quality assessment of photovoltaic modules through electroluminescence images (Luo et al., 2019). In the biomedical field a similar modification towards a conditional PGGAN has yielded improvements for brain metastases detection in magnetic resonance images (Odena et al., 2017). One Class GAN (OCGAN) and Multi-modal One Class GAN (MMOCGAN) presented in Xiong et al. (2020) are an attempt to cope with statistically non-meaningful defect classes by generating samples from the complementary distribution of the "good" class. Reinforcement Learning (RL) methods have also been used to guide data generation and increase the intra-class variability of the generated data, An example is the Actor-Critic GAN (AC-GAN) (Huang et al., 2019), which aims to identify sub-classes from a given class in a preprocessing step and then use Actor-Critic RL on top of the GAN to adjust loss weighting so that augmentation of each sub-class is either encouraged or inhibited. Finally, enabling generation for even smaller datasets is the BigGAN (Brock et al., 2019) adaptation method described in Noguchi and Harada (2019), which proved useful for few-shot learning in Satoshi Tsutsui and Yanwei Fu (2019) and, though still untried in defect detection, served as a major inspiration for our work.

2.1.3 Encoder/decoder architectures

A common type of customized architecture is one based on encoder/decoder approaches to generation. For example, Tang et al. (2020) uses an improved combination of similar encoder/decoder-based generators, namely BEGAN (Berthelot et al., 2017) and Skip-GANomaly (Akçay et al., 2019). Defect-GAN (Zhang et al., 2021) copes with the lack of defect data by synthesizing defects through unpaired image-to-image translation, thus creating additional defects using good images. Its encoder/decoder architecture corresponds to a defacement and restoration process and makes use of a spatial and categorical control map as well as the injection of adaptive noise to increase image diversity. A similar image-to-image translation idea is implemented in the surface defect-generation adversarial network (SDGAN) (Niu et al., 2020) and in Li et al. (2021) which is built around CycleGAN.

A recent and well-performing approach, DeepSMOTE (Dablain et al., 2022), tries to mimic vector-based oversampling approaches but on the level of raw images. It uses an encoder-decoder architecture to produce linear interpolations in the image space similar to SMOTE (Chawla et al., 2002). Although the above method was not used for defect classification, it served as inspiration for our approach of performing oversampling on the image level, which we further adapted to the defect classification problem.

While the proposed methods facilitate both high-fidelity image generation from limited data and targeted oversampling of important inputs, the approach introduced in this paper aims to combine the two leading to a more efficient and less computationally intensive oversampling method performed at the image level.

2.2 Prediction confidence in deep neural networks

As we saw in the previous sections, DNNs and especially convolutional ones, are a powerful learning model. This has come at a cost, however, as the growing model complexity of neural networks - which is also the cause of their better test accuracy - introduces more overconfidence in their predictions (Guo et al., 2017). In this section, we focus on the problem of classification. One way to define confidence in a classification setting is as the maximal value of the last softmax layer, which determines the class of a given input. Comparing this with the validation accuracy of the network for a given class using a reliability diagram for different confidence ranges and their corresponding accuracy scores, Niculescu-Mizil and Caruana (2005) found a significant difference in the 110-layer ResNet compared to the better calibrated but more primitive five-layer LeNet model on the CIFAR-100 dataset.

The main reason for this increasing miscalibration due to increasing model complexity is that DNNs additionally suffer from a more subtle case of overfitting. Namely, they tend to overfit the negative log-likelihood loss invisibly. In contrast, their visible generalization accuracy measured by a 0/1 loss seems to remain stable. This is a sign of unreliability that has limited DNN use in real-world safety-critical applications.

Many methods have been proposed to counter prediction overconfidence. The first category of calibration methods tries to adjust softmax outputs as a post-processing step to resemble the actual confidence probabilities or follow an ordering where a higher value will correspond to higher true confidence. Histogram Binning (Zadrozny & Elkan, 2002), Isotonic Regression, and Bayesian Binning Quantiles (BBQ) (Naeini et al., 2015) are example methods that solve optimization problems after the model training to bring softmax output close to their confidence values as estimated on a validation set. Platt Scaling (Platt, 2000) and its generalizations Matrix Weighting (Guo et al., 2017), and Temperature Scaling (Hinton et al., 2015) are applied on the logit layer just before the softmax aiming to calibrate the weights of the final layer so that outputs are close to the validation set confidence probabilities. Temperature scaling is the most popular approach, as it has the benefit of not influencing the ordering of the class predictions and therefore guaranteeing the exact class prediction as before.

A further category of confidence assessment methods tries to make changes to the learning algorithm so that the training process is constrained to output reasonable measures of the model's true confidence. Most notable is the addition of a penalty term to the loss function that discourages ordering inconsistencies in the output pseudoprobabilities (Moon et al., 2020). Finally, regularization techniques such as dropout, weight decay, label smoothing (Müller et al., 2019) and mixup (Thulasidasan et al., 2019) have also been shown to improve confidence estimates.

Accurately quantifying the prediction confidence of Deep Neural Networks plays an important role in our approach since it helps us determine which samples need to be reinforced through data augmentation. As we are less interested in obtaining probabilistic estimates of confidence and also want to avoid risking a deterioration of the classifier's performance by treating the model as a black box, we focus on a less invasive method introduced for approximating the distance to the classification boundary (Elsayed et al., 2018), which does not require any changes in the network's architecture or the way it is trained.

3 Methods

This work introduces an oversampling method that is applied directly to raw images. The rationale for our approach is that we want to perform oversampling in a way that is decoupled from deep feature extraction, making it possible to train the final classifier end-to-end on the augmented dataset. It can be seen as a method similar to Borderline-SMOTE (Han et al., 2005) focusing on samples close to the classification boundary, but on the level of raw images. Aiming to generate images that are most informative for the way the classifier separates between classes, we rank images according to initial classifier confidence and use low-confidence ones to guide our generation process.

Figure 1 depicts an overview diagram for our proposed approach. It consists of an initial pre-training stage performed on the original imbalanced dataset. The resulting weights are used for the estimation of the boundary between classes and the ranking of instances according to model confidence. After the most informative instances have been selected from the original dataset they are used as seeds for an instance-based

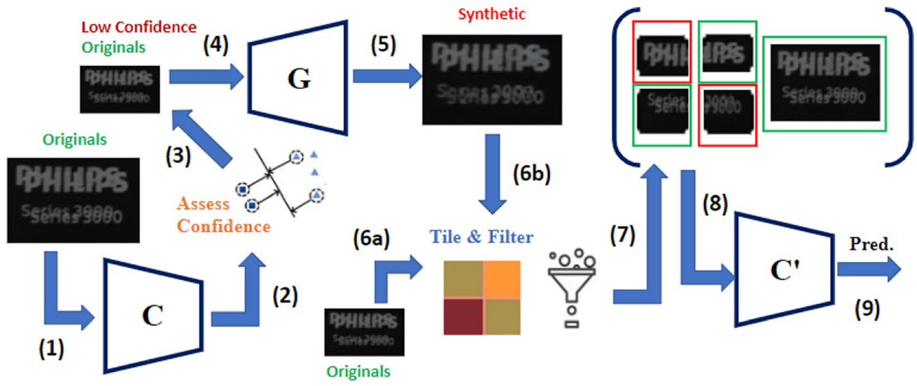

Fig. 1 Basic components and dataflows for the proposed oversampling approach. The sequence of processing steps is outlined with numbers from (1) to (9)

generator, which produces similar images introducing small variations. After post-processing (tiling and fusion with original images) and filtering of sub-standard quality images, we use the generated data to augment the original dataset. The training of the classifier is completed by fine-tuning the weights of the pre-trained classifier using the newly augmented, balanced dataset.

In the following subsections, we provide more details on the Synthetic Image Generation and Confidence Assessment components before fitting everything together to the final oversampling process.

3.1 Synthetic image generation

Producing high-fidelity images for fine-grained classification is challenging, however, state-of-the-art networks such as BigGAN (Brock et al., 2019) or StyleGAN (Karras et al., 2021) have been able to achieve it. Of course, both consist of millions of learnable parameters and require vast training datasets along with the corresponding computational resources. Instead of training such a model from scratch, we make use of a technique inspired by Noguchi and Harada (2019) and Satoshi Tsutsui and Yanwei Fu (2019), which aims to perform transfer learning on a pre-trained BigGAN on ImageNet. BigGAN's generator G is isolated from the discriminator and its weights are initialized to the values obtained from ImageNet. Then for each input image I in the dataset, it is fine-tuned to produce an image I_z, as similar as possible to the original given a random noise vector z as input. This fine-tuning includes only the relearning of the scale and shift parameters of the batch normalization layers. Intuitively this corresponds to selecting only the features relevant to the target dataset from a super-set of features learned through pre-training on ImageNet. The loss function for the fine-tuning is as follows:

$$\mathcal{L}_G(G, I_z, z) = \mathcal{L}_1(G(z), I_z) + \lambda_p \mathcal{L}_{perc}(G(z), I_z) + \lambda_z \mathcal{L}_{EM}(z, r) \tag{1}$$

$\mathcal{L}_1$ is the L1 distance and $\mathcal{L}_{EM}$ the earth mover distance, which tries to regularize z as a Gaussian sample ($r \sim N(0, 1)$); $\mathcal{L}_{perc}$ is the perceptual loss and λ_p, λ_z are regularization coefficients. Finally, to generate multiple images from input I, some random noise is added to the input so that $I_z = G(z + \epsilon)$.

In Algorithm 1 we use the aforementioned generator as an instance-based generator that allows us to produce small variations of an input image. In practice, we observed that it usually produced high-quality defect images. To address the cases where it didn't we added additional quality enhancement measures. The most important of those is provided by the *TilePermutations* function, whose aim is to produce hybrid images by splitting its inputs into halves and quadrants and producing all possible combinations of the split parts (without of course changing their position in the original images). The resulting hybrid images together with the synthetic images are more populous than the n_{aug} images we need per base image. For this reason, we store all synthetic and hybrid images in a min-heap M from which we pick the top n_{aug} images with the lowest mean squared error (MSE) compared to the originals.

Algorithm 1 Generate Synthetic Data

```
Input: $G, \mathcal{L}_G$ image generator and loss, $I_b$ base images, $n_{gen}$ aug. target
Output: $I_{out}$ set of $|I_b| \cdot n_{gen}$ generated images
 1: $I_{out} \Leftarrow \{\}$
 2: $M \Leftarrow MinHeap()$
 3: for $i \in I_b$ do
 4:     for $n \in range(n_{gen})$ do
 5:         $z \Leftarrow \mathcal{U}^{n_z}(0,1)$
 6:         $i_g \Leftarrow G(i, z, \mathcal{L}_G)$
 7:         $I_p \Leftarrow TilePermutations(i, i_g, \{2,4\})$
 8:         for $p \in I_p$ do
 9:             $M \Leftarrow M \cup \{(p, mse(p,i))\}$
10:         end for
11:     end for
12:     for $k \in range(n_{gen})$ do
13:         $m_k, l_k \Leftarrow \underset{m \in M}{argmin}[mse(m,i)]$
14:         $M \Leftarrow M \setminus \{(m_k, l_k)\}$
15:         $I_{out} \Leftarrow I_{out} \cup \{m_k\}$
16:     end for
17: end for
```

3.2 Confidence assessment

To determine which defect instances the classifier is most uncertain of, and can thus benefit from seeing more similar examples of, the approximation of the distance to the classification boundary is the most straightforward approach. Of course, confidence cannot be viewed as a probability, but the relative ordering between distances together with a threshold can give us a limit of the model's knowledge boundaries. Contrary to SVMs, determination of the margin in deep neural networks is a challenging problem, nevertheless (Elsayed et al., 2018) suggests the following approximation, which is used in their calculation of the Large Margin Loss.

The decision boundary between classes i and j is defined as the set of inputs for which the confidence for two classes is equal, f being the (confidence) output of the NN:

$$D_{\{i,j\}} \triangleq \{x \mid f_i(x) = f_j(x)\}$$

The distance of a point x to the decision boundary is then defined under an l_p norm as the smallest displacement of the point that results in confidence equality:

$d_{f,x,\{i,j\}} \triangleq \min_{\delta} \|\delta\|_p \text{ s.t } f_i(x+\delta) = f_j(x+\delta)$

The above optimization problem is intractable for a non-linear f, therefore using the 1st order Taylor approximation to linearize f they obtain the following final approximation for the distance to the margin:

$$\hat{d}_{f,x,\{i,j\}} = \frac{|f_i(x) - f_j(x)|}{\|\nabla_x f_i(x) - \nabla_x f_j(x)\|_q} \tag{2}$$

3.3 On-the-fly image-level oversampling

Algorithm 2 incorporates the outcomes as per the previous sections into the training process. The inputs are a CNN architecture C and the training data (X, Y), as well as the instance-based GAN G, adapted from BigGAN according to Noguchi and Harada (2019) with a loss $\mathcal{L}_G$. Further parameters include n_p which is the number of pre-training epochs to get sufficiently updated weight values to assess model confidence and n the number of epochs to train on the full augmented dataset. k_{top} indicates the number of most informative images selected to serve as seeds for the generation process.

Algorithm 2 On-the-fly Image-Level Oversampling

```
Input: $C$ the CNN, $X, Y$ train data, $G, \mathcal{L}_G$ the GAN and its loss function, $k_{top}$ size of generation base, $n_p, n$ pre-train and train epochs, $l_g$ the label for the good class, $L_d$ the set of defect class labels
Output: $C''$ the trained classifier after oversampling
1:  $M \Leftarrow MinHeap()$
2:  $C' \Leftarrow train(C, X, Y, n_p)$
3:  $X_{good}, Y_{good} \Leftarrow \{(x_i \in X, y_i \in Y) : y_i = l_g\}$
4:  $X_{defect}, Y_{defect} \Leftarrow \{(x_i \in X, y_i \in Y) : y_i \in L_d\}$
5:  for $x \in X_{defect}$ do
6:      $\hat{d} = \frac{|C'_{good}(x) - C'_{defect}(x)|}{\|\nabla_x C'_{good}(x) - \nabla_x C'_{defect}(x)\|_\infty}$
7:      $M \Leftarrow M \cup \{(x, \hat{d})\}$
8:  end for
9:  $I_b \Leftarrow \{\}$
10: $n_{aug} \Leftarrow \lfloor \frac{|Y_{good}| - |Y_{defect}|}{k_{top}} \rfloor$
11: for $k \in range(k_{top})$ do
12:     $x_k, d_k \Leftarrow \underset{x \in M}{argmin}(\hat{d})$
13:     $M \Leftarrow M \setminus \{(x_k, d_k)\}$
14:     $I_b \Leftarrow I_b \cup \{m_k\}$
15: end for
16: $X_{aug}, Y_{aug} \Leftarrow generate(G, \mathcal{L}_G, I_b, n_{aug})$
17: $C'' \Leftarrow train(C', X \cup X_{aug}, Y \cup Y_{aug}, n)$
```

After pre-training for n_p epochs, the distance to the boundary for each training image is computed according to Eq. 2. As expected, this approximation does not provide good results for all images but it works well for images close to the class boundary assigning them smaller values than clearly classified images that are away from the boundary. Data instances and their confidence scores are stored in a min-heap out of which the

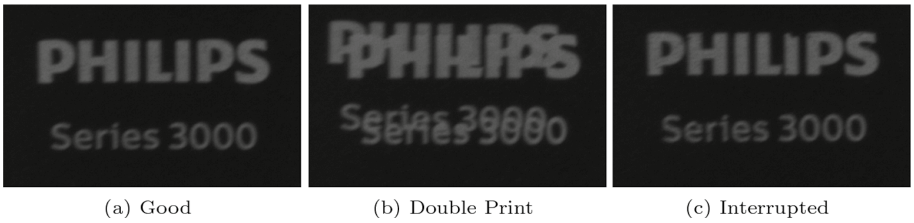

Fig. 2 Original Shaver Shell Prints

k_{top} lowest distance images are extracted to form the base set for the generation. After determining the number of synthetic images to be generated per base image, needed for rebalancing the dataset, we pass the base images to the generation process described in Algorithm 1. The parameter n_{aug} defines the number of images to be generated per selected base image so that the final dataset is balanced between defects and non-defects and is determined by the integer division of the difference between the number of images in the good class $|Y_{good}|$ and the number of total defective images $|Y_{defect}|$ over the number of base-images k_{top}. Note that the number of generated images per individual defect class might differ; there is only a constraint that the total defects are balanced with the good images. Depending on how many low-confidence images a defect class has, the more it needs to be augmented according to our approach. Following the augmentation step, the pre-trained classifier is trained for a further n epochs to produce a better classification boundary.

4 Results

Throughout our experiments, we show how the presented oversampling method benefits the general defect classification problem, by comparing it both with state-of-the-art approaches used in defect datasets and image- and vector-level oversampling approaches. We performed our experiments using a dataset of shaver shell logo print images from a real production line presented in the section below.

4.1 Dataset information

The dataset used was provided by Philips Consumer Lifestyle B.V. and was collected from their pad printing process to serve the need for building an automated quality inspection system. As described earlier, owing to the infrequency of defects in their process, it was hard to gather many defect images leading to an imbalanced dataset.

The dataset consists of JPEG RBG images with dimensions 220 × 360. They are divided into three classes, one good and two defect classes, namely double prints and interrupted prints. Representative examples of each class are presented in Fig. 2. The number of correctly printed images is 2684, of double prints 244, and of interrupted prints 598. One important feature to note is that interrupted prints can be very similar to good prints, making their distinction difficult, as well as the generation of sufficiently differentiated images from these two classes.

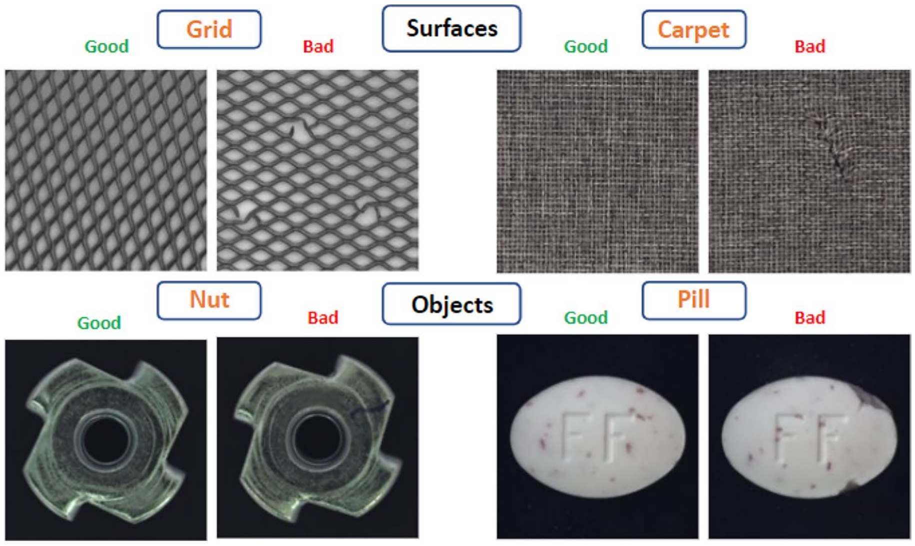

Fig. 3 Samples from the MVTec AD datasets

Table 1 Number of class instances for the Shavers and MVTec AD product datasets including train and test sets

	Train		Test		Total	
Datasets	Good	Defects	Good	Defects	Good	Defects
Shavers	2147	674	537	168	2684	842
Grid	211	46	53	11	264	57
Carpet	224	71	56	18	280	89
Metal Nut	176	74	44	19	220	93
Pill	214	113	53	28	267	141

Moreover, to verify the robustness of our method we used four additional datasets of product defects from the MVTec AD collection (Bergmann et al., 2019). This is a collection of datasets consisting of surface and object defects. For our evaluation, we chose two products from each category that exhibited similar defects to the shavers dataset leading to the high similarity between classes. From the surfaces, we used the carpet and grid datasets and from the objects the pill and metal nut datasets, samples of which are shown in Fig. 3.

Table 1 shows the number of instances belonging to each class for all datasets used as well as their train and test set sizes as determined by the 5-fold cross-validation scheme described in the next section.

4.2 Experimental setup

Our experimental process was designed to compare our approach with three other families of approaches that have been common in the literature. The first is the attempt to directly generate data of the highest fidelity possible using a powerful generation method. We use StyleGAN as a comparison which achieved good results in Achicanoy et al. (2021). The second type is the use of transfer learning and namely Resnet50 used in many works such

Table 2 Comparison of oversampling methods on the shaver-shell prints dataset

Method	Bin. Recall %	AUROC %	Precision %	F1 %
Resnet50	85.85 ± 1.50	98.85 ± 0.12	94.41 ± 3.27	89.59 ± 1.27
Resnet50+SMOTE	95.84 ± 0.52	98.87 ± 0.13	84.53 ± 3.01	89.61 ± 1.57
Resnet50+ADASYN	95.49 ± 0.99	99.07 ± 0.11	85.14 ± 3.45	89.67 ± 1.69
Custom CNN	95.84 ± 0.39	99.20 ± 0.19	97.53 ± 0.81	96.67 ± 0.56
Custom CNN+LW	96.07 ± 0.39	99.09 ± 0.19	98.34 ± 0.33	**97.19 ± 0.43**
StyleGAN	91.20 ± 2.20	99.01 ± 0.14	**99.17 ± 0.41**	94.95 ± 1.38
DeepSMOTE	93.58 ± 1.07	99.23 ± 0.15	96.93 ± 0.80	95.22 ± 0.87
Ours	**97.27 ± 0.76**	**99.34 ± 0.07**	96.82 ± 1.27	97.03 ± 0.98

The scores of the best-performing approaches for each experimental metric is shown in bold

as Zhang et al. (2021) and Feng et al. (2021) for transfer learning, also viewing it in combination with vector-based oversampling. Thirdly we compare against DeepSMOTE (Dablain et al., 2022), a state-of-the-art approach of performing SMOTE-like oversampling on the image level. For the non-transfer learning scenarios, we used as a classifier a customized shallow CNN for this dataset consisting of a convolutional layer with two parallel filters of $(3 \times 3) \times 16$ and $(1 \times 1) \times 16$ followed by a dense and a softmax layer.

The metric monitored was the binary recall from the perspective of the defect classes (Table 2), i.e. the defect class is considered the positive class for measuring recall. We found this metric most appropriate for a defect classification example as it better suits the way automated visual inspection is envisioned to work on a real production line. More specifically, positive predictions (good) usually receive the green light with no or little manual checking, while negative ones (defects) are put aside to be further examined by a human operator. Our aim is to minimize the number of defects that are mistakenly labeled as high-quality products. A more precise definition of Binary Recall, as used in the current context, can be formulated given the classifier C, test data X, the labeling function l, and the set of defect labels $L_d = \{double\,print, interrupted\}$, as follows:

$$BinaryRecall = \frac{|x \in X : C(x) \in L_d \wedge l(x) \in L_d|}{|x \in X : l(x) \in L_d|}$$

Another benefit of this metric is that it does not suffer from the dataset skew as, for example, accuracy which is dominated by the accuracy in the majority class. One must also be careful while maximizing binary recall so that not every product image is classified as a defect. For this reason, we also evaluated the ROC-AUC score which measures class separability, though with a relative skew towards the majority class. The ROC-AUC score was satisfying for all experiments with values greater than 98%.

To complete the picture of the final classifiers' performance, we include the binary Precision and F1-scores, again, measured from the perspective of the defect class. Precision performance will be determined by the percentage of good images that get mistakenly classified as defects, while the F1-score will attempt to give a balanced account of the methods' effects on precision and recall. More precisely, these metrics are calculated using the same notation as for Binary Recall as follows:

$$Precision = \frac{|x \in X : C(x) \in L_d \wedge l(x) \in L_d|}{|x \in X : C(x) \in L_d|}$$

$$F1 - score = \frac{2 \times Precision \times BinaryRecall}{Precision + BinaryRecall}$$

Additionally, we compare a simplistic augmentation using our generation method in an untargeted fashion against our targeted oversampling approach based on the selection of the most informational examples (Fig. 6). This helps us gain further insight into how and in which cases targeted oversampling is helpful. We also monitor additional metrics such as the number of images generated for each defect class and the class-specific recalls.

The experiments consist of a total of 30 model runs using 5-fold cross-validation on a single NVidia K80 GPU used for both the training and data generation processes. Binary Recall scores are presented with their 95% confidence intervals.

4.2.1 Hyperparameter tuning

For most comparison methods, an exhaustive search was carried out over ranges around initial well-performing hyperparameters (HPs) determined through trial and error. The best-performing hyperparameters were chosen over a stratified 5-fold cross-validation scheme similar to that followed for the showcased experiments resulting in an overall nested cross-validation (or double-cross) scheme as described in Stone (1974) and specified in pseudo-code in Algorithm 3. Specifically, the inner cross-validation produces validation sets for the selection of HPs and the outer cross-validation produces independent test sets for out-of-sample evaluation of the methods with the best-performing HPs. From this scheme, we extract the most frequently selected HP combinations as the recommended set of HPs to use for each method, which could provide the interested reader with insight into the dataset from an oversampling perspective.

Algorithm 3 Nested Cross-Validation for Evaluation and Hyper-Parameter Tuning

Input: C the Classifier (possibly including pre-trained embeddings and/or an oversampling method), X the input data (images), Y the input labels, CV the stratified cross-validation scheme, $\mathcal{H}$ the set of possible hyperparameter combinations, r_s the random seed for the current evaluation run

Output: M the complete final metrics per fold and seed, H_f the selected hyperparameter combinations per fold and seed

```
1:  $M \Leftarrow \{\}$
2:  $H_f \Leftarrow \{\}$
3:  $fold \Leftarrow 0$
4:  for $Train, Test \in CV.split(X, Y, folds = 5, random = r_s)$ do
5:      $H_m = MinHeap()$
6:      for $H \in \mathcal{H}$ do
7:          $fold \mathrel{+}= 1$
8:          $R_{avg} \Leftarrow 0$
9:          for $Train_{HP}, Test_{HP} \in CV.split(Train, 5, r_s)$ do
10:             $C_H \Leftarrow train(C, Train_{HP}, H)$
11:             $m_H \Leftarrow evaluate(C_H, Test_{HP})$
12:             $R_{avg} \Leftarrow R_{avg} + m_H.recall$
13:         end for
14:         $R_{avg} \Leftarrow R_{avg}/5$
15:         $H_m \Leftarrow H_m \cup \{(H, R_{avg})\}$
16:     end for
17:     $h_{top}, r_{top} \Leftarrow \underset{(h,R) \in H_m}{argmax}\,[R]$
18:     $C_f \Leftarrow train(C, Train, h_{top})$
19:     $m \Leftarrow evaluate(C_f, Test)$
20:     $H_f \Leftarrow M \cup \{(r_s, fold, h_{top})\}$
21:     $M \Leftarrow M \cup \{(r_s, fold, m)\}$
22: end for
```

4.3 Experimental results

As shown in Table 2 our method outperforms all state-of-the-art approaches in terms of binary recall. However, there are several interesting points to note. Firstly, we observe that the custom CNN architecture outperforms transfer learning and transfer learning with oversampling approaches because the features are learned end-to-end specifically for the dataset at hand instead of being adapted from imagenet. Secondly, the impact of oversampling on the vanilla Resnet50 approach is much larger, than the effect of both Loss Weighting and

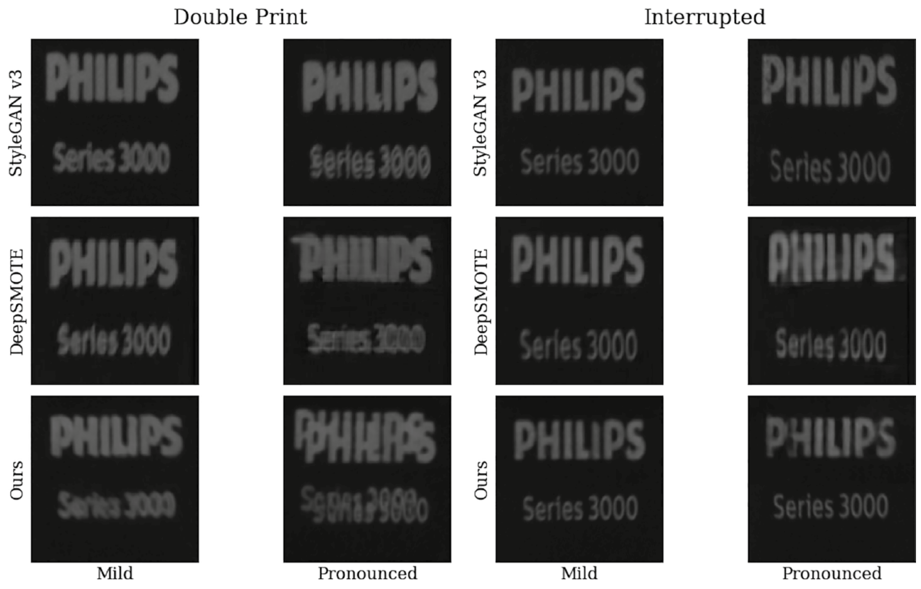

Fig. 4 Artificially generated defect images

our approach on the custom CNN. This can be attributed to greater margins for improvement in lower recalls, but also to the imperfection of the generation methods at the image level, which is a much more complicated, high-dimensional process than generating simple vectors.

Most interestingly, we observed that the augmentation approaches based on StyleGAN and DeepSMOTE had an adverse effect on the custom CNN's performance. This is mainly attributed to their inability to produce realistic defect images that are close but not identical to the high-quality images and can also be hinted at by the samples of generated images shown in Fig 4. In fact, on the MVTec AD datasets, which are one order of magnitude smaller in size, these generative methods failed to produce plausible defect images, most probably due to the documented early overfitting approach of GAN architectures on small datasets (Karras et al., 2020). Therefore they are also not included in Table 3. Our generator, thanks to the additional processing steps introduced manages to usually depict these kinds of small defects, which occur mostly in the interrupted class of the shaver dataset. Nevertheless, confusing synthetic images were still occasionally produced in some of the dataset's splits leading to a small deterioration in performance, highlighting a possible limitation of the proposed method.

It is important to note that in terms of AUROC, our method does not provide a significant improvement as it does with binary recall. The purpose of monitoring the AUROC metric, as mentioned in the experimental results section, is to ensure that while our method improves recall in the defect classes, it does not, at the same time, significantly sacrifice performance in the good class. Let us also note that since AUROC considers the dataset as a whole it makes it difficult for improvements in recall to be reflected since they are overshadowed by the performance in the majority class, which is more similar across the different methods.

Of particular interest is the effect of the proposed method on the Precision and F1 metrics in this dataset of high inter-class similarity. As explained in the experimental

Table 3 Comparison of oversampling methods on the MVTec AD product datasets

Dataset	Method	Binary recall %	AUROC %	Precision %	F1 %
Grid	Resnet50	30.30 ± 5.59	73.29 ± 3.48	42.4 ± 5.57	34.36 ± 5.95
	Resnet50 + SMOTE	35.60 ± 5.71	74.67 ± 4.34	48.27 ± 3.38	43.55 ± 3.48
	Resnet50 + ADASYN	42.57 ± 8.38	74.26 ± 4.04	42.93 ± 4.59	35.93 ± 6.68
	Custom CNN	70.90 ± 10.93	90.71 ± 5.49	80.23 ± 9.97	74.50 ± 10.50
	Custom CNN + LW	69.24 ± 12.25	89.80 ± 6.09	75.38 ± 12.3	71.55 ± 12.15
	Ours	71.21 ± 9.92	91.22 ± 5.12	91.43 ± 6.86	78.45 ± 8.36
Carpet	Resnet50	81.89 ± 3.70	97.07 ± 0.41	87.80 ± 3.33	84.20 ± 2.27
	Resnet50 + SMOTE	88.69 ± 1.53	97.21 ± 0.46	79.56 ± 2.11	83.66 ± 0.94
	Resnet50 + ADASYN	84.18 ± 2.71	97.25 ± 0.45	83.96 ± 3.78	83.42 ± 1.28
	Custom CNN	87.77 ± 7.62	98.94 ± 0.49	89.73 ± 1.23	87.48 ± 4.75
	Custom CNN + LW	91.11 ± 6.06	98.90 ± 0.51	88.02 ± 1.78	88.92 ± 3.72
	Ours	92.22 ± 3.32	99.86 ± 0.11	92 ± 1.60	91.9 ± 1.97
Metal Nut	Resnet50	84.03 ± 3.46	96.90 ± 0.79	95.33 ± 1.70	88.99 ± 1.97
	Resnet50 + SMOTE	88.30 ± 3.71	97.32 ± 0.51	90.32 ± 1.33	89.07 ± 1.62
	Resnet50 + ADASYN	84.09 ± 3.71	97.01 ± 0.72	95.38 ± 1.63	89.02 ± 2.09
	Custom CNN	82.92 ± 5.36	97.49 ± 1.15	98.33 ± 1.33	89.55 ±3.87
	Custom CNN + LW	82.92 ± 5.36	97.49 ± 1.15	98.33 ± 1.33	89.55 ± 3.87
	Ours	92.63 ± 3.15	98.32 ± 1.22	98.75 ± 1.00	95.49 ± 2.12
Pill	Resnet50	71.52 ± 6.29	92.70 ± 1.63	84.84 ± 1.63	76.65 ± 4.29
	Resnet50 + SMOTE	90.02 ± 2.62	91.76 ± 1.82	60.7 ± 1.57	72.34 ± 1.41
	Resnet50 + ADASYN	78.62 ± 4.16	91.87 ± 1.70	82.29 ± 1.54	80.08 ± 2.41
	Custom CNN	88.71 ± 2.18	98.35 ± 0.60	93.48 ± 2.03	90.94 ± 1.76
	Custom CNN + LW	88.71 ± 2.18	98.35 ± 0.60	93.48 ± 2.03	90.94 ± 1.76
	Ours	92.29 ± 3.79	98.80 ± 0.58	96.25 ± 1.63	94.11 ± 2.68

results section, our on-the-fly oversampling method was designed to optimize recall, which in the case of datasets with high inter-class similarity might come at the expense of precision i.e. mistakenly classifying more good images than before as defects. This is also illustrated by the method's performance in the precision metric which is lower than all other methods utilizing a custom CNN classifier. Consequently, its F1-score, while second highest, is overcome by the Custom CNN with Loss Weighting. The sacrifice of the F1 metric, however, is only 0.16, with largely overlapping confidence intervals between the two methods, showing a small sacrifice in the overall problem performance.

On the MVTec AD datasets, confidence-aware oversampling managed to provide the biggest improvements upon the end-to-end trained network, achieving the best recall scores in all cases. However, establishing statistical significance through confidence intervals was harder in this case, due to the very low number of defects in the test sets (see Table 3). As a consequence of the low number of defect samples, mispredicting just a few images has a pronounced impact on the overall binary recall score, which unfortunately presents a limitation of the evaluation scheme of our method when faced with smaller minority classes. Still, the improvement in the Metal Nut dataset was significant in comparison with most methods, while in the pill and carpet datasets, there are

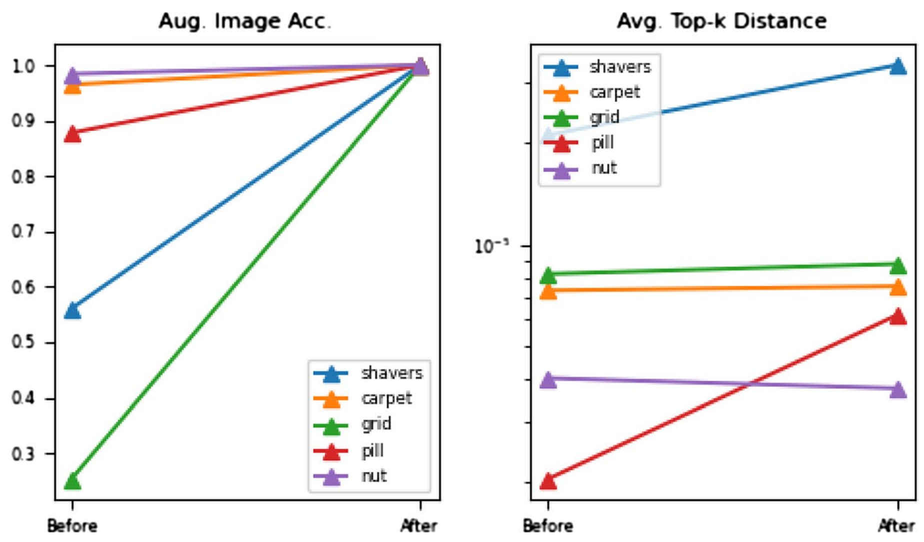

Fig. 5 Label accuracy of augmented images, before and after augmented training (Left). Top-k distances to classification boundary before and after augmented training for k = 15 (Right)

some indications of improvement. The grid dataset was harder for all methods producing results with very high variability between individual run scores.

Contrary to the shaver's dataset performance in the precision and F1 scores is consistently the highest across the four MVTec AD products - in the Metal Nut and Pill datasets being also statistically significant. We attribute this difference in the corresponding performances as measured on the original Shavers dataset, again, to the smaller amount of data which benefits significantly from the addition of the augmented images resulting in more precise boundaries from the perspective of both classes. For this reason, our image-level oversampling method has a more global effect on the classifiers' performance, not suffering from the trade-offs appearing in the more populous shavers dataset.

To understand the proposed method in more depth, Fig. 5 shows the changes in classifying augmented images and in the top-15 minority instance distances to the boundary before and after augmentation. There is an indication that boundaries shift from the minority classes closer to the majority classes so that generated images that were misclassified before augmented training are now learned by the model. Of course, this shift in the distances is varied and cannot easily be correlated with performance increases, due to the complexity of the deep learning process and the approximate nature of the distance calculation method. It is important to note that distances both before and after augmentation are low in magnitude considering the high dimensionality of the feature space, hinting at the existence of highly populated boundaries. The goal of our method is to push those boundaries slightly so that they are biased toward the minority class—whose recall is more important—while perhaps, as in the case of the Shavers dataset, sacrificing prediction accuracy over the majority class—which is desirable given that the performance sacrifice is limited. This small shift could be significant exactly because the boundaries are densely populated due to high-class similarity. In the case of the MVTec AD datasets, this process leads to an overall improvement of class separability as highlighted by the increases in both precision and recall.

Table 4 Table of searched and recommended final hyperparameters per examined method for the shavers dataset

Method	Searched HP	Recommended HP
SMOTE	$type \in \{None, borderline1, borderline2\}$, $k \in [2, 20], m \in [0, 22]$	$type = borderline2$ $k = 2, m = 20$
ADASYN	$k \in [2, 20]$	$k = 5$
Custom CNN	$batch\text{-}size \in \{4, 8, ..., 64\}$, $l_r \in \{10^{-5}, 10^{-4}, ..., 10^{-2}\}$, $dropout \in \{0.2, 0.3, ..., 0.8\}$	$batch\text{-}size = 4$ $l_r = 10^{-4}$ $dropout = 0.4$
Ours	$top\text{-}k \in [5, 50]$, $pre\text{-}eps \in \{5, 10, .., 45\}$	$top\text{-}k = 15$ $pre\text{-}eps = 20$

In terms of the specific hyperparameters (HPs) of our approach defined as inputs to Algorithm 2, after comparing the final classification performance of different combinations we chose 20 epochs for pretraining and 30 epochs with early stopping for training on the augmented dataset as the best-performing way to split the 50 total epochs needed to reach a stable loss plateau. We also determined the best value for k_{top} to be 15 images. In all other approaches, the training epochs for the classifier were 50 with early stopping, so that all comparison classifiers have time to reach their loss plateaus and equal to the total amount of training epochs used in our approach. The number of augmentation examples produced for the comparison methods was always the required amount for every class to have as many instances as the good class, resulting in a balanced dataset. The ranges of HPs examined and the final recommended HPs for the Shavers dataset are shown in Table 4. For the image generation of StyleGAN and DeepSMOTE, we used the settings suggested for small datasets in the respective papers (Karras et al., 2021; Dablain et al., 2022).

Finally, in terms of computation time, the introduced method was much quicker by approximately 3× the training time without augmentation (∼ 30 mins for a full run), while other image-level approaches such as StyleGAN and DeepSMOTE took more than 20 h to train. This is because our method uses a small base set of images for generation and the time taken is linearly proportional to the number of base images. It is also built on top of a lightweight transfer learning method for GANs, while DeepSMOTE needs to be trained from scratch and StyleGAN's fine-tuning is more time-consuming due to its vast number of parameters.

Figure 6 shows more closely how our oversampling method helps the classifier's learning process in the shavers dataset. We compare binary and class-specific recalls by using our generation method in a uniform way with the whole training set as seeds and selecting the seed set based on a distance-to-boundary confidence measure. What stands out is that the majority of the images close to the decision boundary belong to the interrupted class which is most similar to the good class. Basing the augmentation off of those images is also what brings the largest gains in recall performance. In the double print category, such gains are not visible, in fact, performance slightly deteriorates. This hints at a limitation of our method consistently producing performance gains over a range of imbalanced learning scenarios as it has been primarily designed for problems with high inter-class similarity.

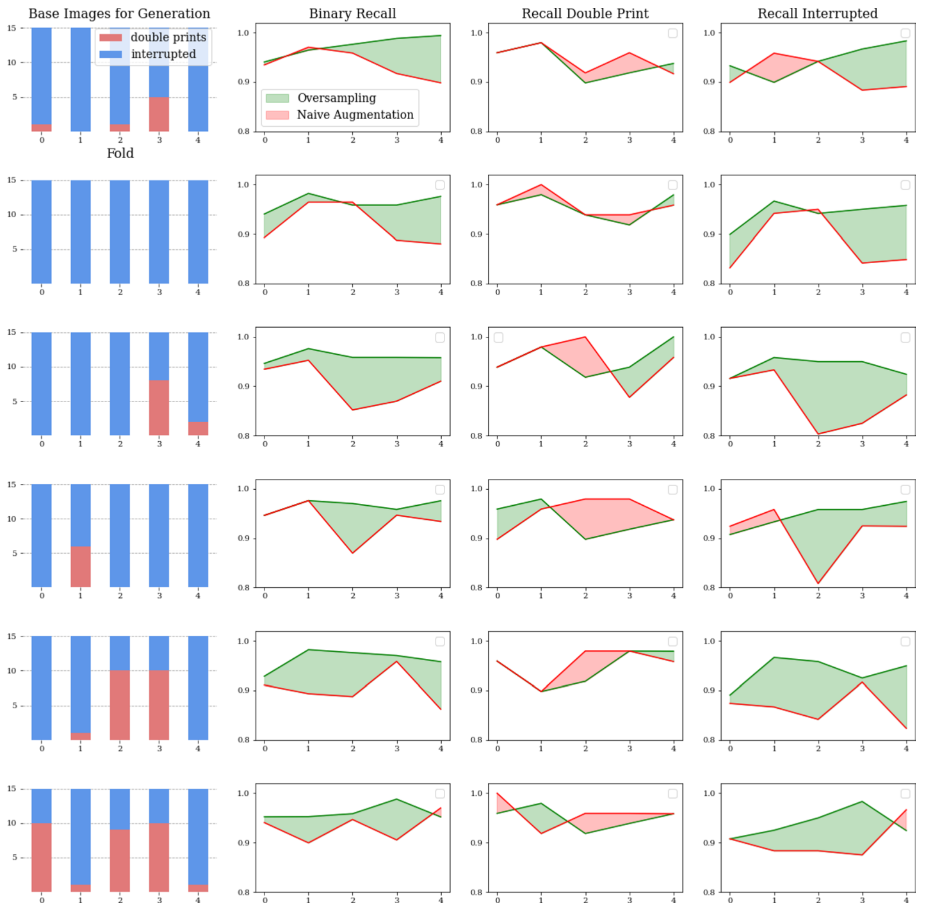

Fig. 6 Comparison between simple augmentation and confidence-based oversampling—6 different instances of 5-fold CV on the shavers dataset

5 Conclusion

In this work, we introduced a novel method for performing oversampling at the image level in the context of defect detection. Data generation is now performed more efficiently based on images that are estimated to be close to the classification boundary. The high-fidelity images generated helped improve the classification results over a dataset containing defects of varying perceptibility. The runtime and computational costs of generating synthetic data were also greatly reduced compared to other state-of-the-art approaches.

We believe that future advances in instance-based or few-shot image generation can greatly help improve our work by producing images of higher fidelity and variability from a small selected seed set of low-confidence images. Further opportunities for improvement lie in the way original and synthetic images are fused, which could potentially be performed in a smoother way than tiling using a few-shot learning-based fusion method. Finally, it is worth investigating how to produce linear interpolations between low-confidence samples through a suitable encoder/decoder architecture.

Acknowledgements We would like to thank Philips Consumer Lifestyle BV for the data provided.

Funding Open access funding provided by HEAL-Link Greece. This work was supported by the European Union's Horizon 2020 program project STAR under grant agreement number H2020-956573.

Data availability All data used is open-source except for the complete dataset of logo prints and defects, which was provided confidentially by Philips Consumer Lifestyle BV and is proprietary. However, a similar synthetic dataset can be provided.

Code availability Code implementation is available at: https://github.com/tspyrosk/oversampling-defect-recognition

Declarations

Conflicts of interest The authors declare that there is no conflict of interest.

Ethics approval Not Applicable

Consent to participate Not Applicable

Consent for publication All authors who participated in this study give the publisher the permission to publish this work.

References

Achicanoy, H., Chaves, D., & Trujillo, M. (2021). Stylegans and transfer learning for generating synthetic images in industrial applications. *Symmetry*. https://doi.org/10.3390/sym13081497

Akçay, S., Atapour-Abarghouei, A. & Breckon, T. (2019). Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection. In *2019 International Joint Conference on Neural Networks (IJCNN)*, 1–8

Bergmann, P., Fauser, M., Sattlegger, D. & Steger, C. (2019). Mvtec ad - a comprehensive real-world dataset for unsupervised anomaly detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9584–9592. https://doi.org/10.1109/CVPR.2019.00982

Berthelot, D., Schumm, T. & Metz, L. (2017). Began: Boundary equilibrium generative adversarial networks. arXiv:1703.10717

Bird, J. J., Barnes, C. M., Manso, L. J., Ekart, A., & Faria, D. R. (2022). Fruit quality and defect image classification with conditional GAN data augmentation. *Scientia Horticulturae, 293*, 110684. https://doi.org/10.1016/j.scienta.2021.110684

Brock, A., Donahue, J. & Simonyan, K. (2019). Large scale gan training for high fidelity natural image synthesis. ArXiv **abs/1809.11096**

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority oversampling technique. *J. Artif. Int. Res., 16*(1), 321–357.

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I. & Abbeel, P. (2016). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*

Dablain, D., Krawczyk, B., & Chawla, N. V. (2022). Deepsmote: Fusing deep learning and smote for imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*. https://doi.org/10.1109/TNNLS.2021.3136503

Dalal, N. & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893. Ieee

Elsayed, G.F., Krishnan, D., Mobahi, H., Regan, K. & Bengio, S. (2018). Large margin deep networks for classification. arXiv:1803.05598pdf

Fathy, Y., Jaber, M., & Brintrup, A. (2021). Learning with imbalanced data in smart manufacturing: A comparative analysis. *IEEE Access, 9*, 2734–2757. https://doi.org/10.1109/ACCESS.2020.3047838

Feng, X., Gao, X., & Luo, L. (2021). A resnet50-based method for classifying surface defects in hot-rolled strip steel. *Mathematics*. https://doi.org/10.3390/math9192359

Frid-Adar, M., Klang, E., Amitai, M., Goldberger, J. & Greenspan, H. (2018). Synthetic data augmentation using gan for improved liver lesion classification. In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 289–293. https://doi.org/10.1109/ISBI.2018.8363576

Goodfellow, I., Bengio, Y. & Courville, A. (2016). Deep Learning. MIT Press. http://www.deeplearningbook.org

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K.Q. (Eds.) *Advances in Neural Information Processing Systems*, vol. 27. Curran Associates, Inc.

Guo, C., Pleiss, G., Sun, Y. & Weinberger, K.Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, vol. 70. ICML'17, pp. 1321–1330. JMLR.org.

Han, C., Murao, K., Noguchi, T., Kawata, Y., Uchiyama, F., Rundo, L., Nakayama, H. & Satoh, S. (2019). Learning more with less: Conditional pggan-based data augmentation for brain metastases detection using highly-rough annotation on mr images. CIKM '19, pp. 119–127. Association for Computing Machinery, New York. https://doi.org/10.1145/3357384.3357890.

Han, H., Wang, W. & Mao, B. (2005). Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *ICIC*

Hinton, G., Vinyals, O. & Dean, J. (2015). Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*. arXiv:1503.02531

Huang, L., Lin, K. C. J. & Tseng, Y. C. (2019). Resolving intra-class imbalance for gan-based image augmentation. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 970–975. https://doi.org/10.1109/ICME.2019.00171

Jain, S., Seth, G., Paruthi, A., Soni, U. & Kumar, G. (2020). Synthetic data augmentation for surface defect detection and classification using deep learning. *Journal of Intelligent Manufacturing*, 1–14

Jiang, Y., Chang, S. & Wang, Z. (2021). Transgan: Two transformers can make one strong gan. arXiv preprint arXiv:2102.07074

Kadar, M. & Onita, D. (2019). A deep cnn for image analytics in automated manufacturing process control. In *2019 11th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pp. 1–5 . https://doi.org/10.1109/ECAI46879.2019.9042159

Karras, T., Aila, T., Laine, S. & Lehtinen, J. (2018). Progressive growing of GANs for improved quality, stability, and variation. arXiv:1710.10196

Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J. & Aila, T. (2020). Training generative adversarial networks with limited data. In *Proc. NeurIPS*

Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J. & Aila, T. (2020). Training generative adversarial networks with limited data. In *Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS'20*. Curran Associates Inc., Red Hook.

Karras, T., Laine, S., & Aila, T. (2021). A style-based generator architecture for generative adversarial networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence, 43*(12), 4217–4228. https://doi.org/10.1109/TPAMI.2020.2970919

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM, 60*(6), 84–90.

Le, X., Mei, J., Zhang, H., Zhou, B., & Xi, J. (2020). A learning-based approach for surface defect detection using small image datasets. *Neurocomputing, 408*, 112–120. https://doi.org/10.1016/j.neucom.2019.09.107

Liu, L., Cao, D., Wu, Y. & Wei, T. (2019). Defective samples simulation through adversarial training for automatic surface inspection. *Neurocomput*. 360(C), 230–245. https://doi.org/10.1016/j.neucom.2019.05.080

Li, D., Xie, W., Wang, B., Zhong, W., & Wang, H. (2021). Data augmentation and layered deformable mask r-CNN-based detection of wood defects. *IEEE Access, 9*, 108162–108174. https://doi.org/10.1109/ACCESS.2021.3101247

Luan, F., Paris, S., Shechtman, E. & Bala, K. (2018). Deep painterly harmonization. Computer Graphics Forum **37**. https://doi.org/10.1111/cgf.13478

Luo, Z., Cheng, S. Y., & Zheng, Q. Y. (2019). GAN-based augmentation for improving CNN performance of classification of defective photovoltaic module cells in electroluminescence images. *IOP Conference Series: Earth and Environmental Science, 354*(1), 012106. https://doi.org/10.1088/1755-1315/354/1/012106

Meister, S., Mueller, N., Stoeve, J. & Groves, R. (2021). Synthetic image data augmentation for fibre layup inspection processes: Techniques to enhance the data set. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-021-01738-7

Moon, J., Kim, J.-h., Shin, Y. & Hwang, S. (2020). Confidence-aware learning for deep neural networks. In *ICML*

Müller, R., Kornblith, S. & Hinton, G. E. (2019). When does label smoothing help? In *NeurIPS*

Naeini, M. P., Cooper, G. F. & Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. AAAI'15*, pp. 2901–2907. AAAI Press.

Niculescu-Mizil, A. & Caruana, R. (2005). Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning. ICML '05*, pp 625–632. Association for Computing Machinery, New York. https://doi.org/10.1145/1102351.1102430.

Niu, S., Li, B., Wang, X., & Lin, H. (2020). Defect image sample generation with GAN for improving defect recognition. *IEEE Transactions on Automation Science and Engineering, 17*(3), 1611–1622. https://doi.org/10.1109/TASE.2020.2967415

Noguchi, A. & Harada, T. (2019). Image generation from small datasets via batch statistics adaptation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2750–2758

Odena, A., Olah, C. & Shlens, J. (2017). Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning*, vol 70. ICML'17, pp. 2642–2651. JMLR.org.

Pawara, P., Okafor, E., Schomaker, L., & Wiering, M. (2017). Data augmentation for plant classification. In J. Blanc-Talon, R. Penne, W. Philips, D. Popescu, & P. Scheunders (Eds.), *Advanced Concepts for Intelligent Vision Systems* (pp. 615–626). Springer.

Peres, R.S., Azevedo, M., Araujo, S.O., Guedes, M., Miranda, F. & Barata, J. (2021). Generative adversarial networks for data augmentation in structural adhesive inspection. *Applied Sciences***11**(7). https://doi.org/10.3390/app11073086

Platt, J. (2000). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv. Large Margin Classif. **10**

Saiz, F.A., Alfaro, G., Barandiaran, I. & Grana, M. (2021). Generative adversarial networks to improve the robustness of visual defect segmentation by semantic networks in manufacturing components. *Applied Sciences***11**(14). https://doi.org/10.3390/app11146368

Sampath, V., Maurtua, I., Aguilar Martín, J. J., & Gutierrez, A. (2021). A survey on generative adversarial networks for imbalance problems in computer vision tasks. *Journal of Big Data, 8*, 1–2. https://doi.org/10.1186/s40537-021-00414-0

Satoshi Tsutsui, D. C. & Yanwei, F. (2019). Meta-reinforced synthetic data for one-shot fine-grained visual recognition. In *Advances in Neural Information Processing Systems (NeurIPS).*

See, J.E. (2012). Visual inspection : a review of the literature. Sandia Report SAND2012-8590, Sandia National Laboratories, Albuquerque, New Mexico

Sohn, K., Lee, H. & Yan, X. (2015). Learning structured output representation using deep conditional generative models. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (Eds.) *Advances in Neural Information Processing Systems*, vol. 28. Curran Associates, Inc.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Methodological), 36*(2), 111–133. https://doi.org/10.1111/j.2517-6161.1974.tb00994.x

Tang, T. W., Kuo, W. H., Lan, J. H., Ding, C. F., Hsu, H. & Young, H. T. (2020). Anomaly detection neural network with dual auto-encoders GAN and its industrial inspection applications. *Sensors***20**(12). https://doi.org/10.3390/s20123336

Thulasidasan, S., Chennupati, G., Bilmes, J., Bhattacharya, T. & Michalak, S. (2019). On mixup training: Improved calibration and predictive uncertainty for deep neural networks. arXiv:abs/1905.11001

Tulbure, A.-A., Tulbure, A.-A., & Dulf, E.-H. (2022). A review on modern defect detection models using DCNNs—deep convolutional neural networks. *Journal of Advanced Research, 35*, 33–48. https://doi.org/10.1016/j.jare.2021.03.015

Viola, P. & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. CVPR 2001, vol. 1, p. IEEE

Wang, C. & Xiao, Z. (2021). Lychee surface defect detection based on deep convolutional neural networks with GAN-based data augmentation. Agronomy **11**(8). https://doi.org/10.3390/agronomy11081500

Wang, Y., Luo, S., & Wu, H. (2021). Defect detection of solar cell based on data augmentation. *Journal of Physics: Conference Series, 1952*(2), 022010. https://doi.org/10.1088/1742-6596/1952/2/022010

Xiong, W., Lee, J., Qu, S., & Jang, W. (2020). Data augmentation for applying deep learning to display manufacturing defect detection. *SID Symposium Digest of Technical Papers, 51*, 1210–1213. https://doi.org/10.1002/sdtp.14096

Yun, J. P., Shin, W. C., Koo, G., Kim, M. S., Lee, C., & Lee, S. J. (2020). Automated defect inspection system for metal surfaces based on deep learning and data augmentation. *Journal of Manufacturing Systems, 55*, 317–324. https://doi.org/10.1016/j.jmsy.2020.03.009

Zadrozny, B. & Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '02*, pp. 694–699. Association for Computing Machinery, New York. https://doi.org/10.1145/775047.775151.

Zhang, H., Chen, Z., Zhang, C., Xi, J. & Le, X. (2019). Weld defect detection based on deep learning method. In *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*, pp. 1574–1579. https://doi.org/10.1109/COASE.2019.8842998

Zhang, G., Cui, K., Hung, T. Y. & Lu, S. (2021). Defect-gan: High-fidelity defect synthesis for automated defect inspection. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2523–2533.

Zhang, G., Cui, K., Hung, T.-Y. & Lu, S. (2021). Defect-gan: High-fidelity defect synthesis for automated defect inspection. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 2523–2533. https://doi.org/10.1109/WACV48630.2021.00257

Zhang, Y., Wa, S., Sun, P., & Wang, Y. (2021). Pear defect detection method based on RESNET and DCGAN. *Information*. https://doi.org/10.3390/info12100397

Authors and Affiliations

Spyros Theodoropoulos[1,2] · **Patrik Zajec**[3] · **Jože M. Rožanec**[3] · **Dimosthenis Kyriazis**[2] · **Panayiotis Tsanakas**[1]

✉ Spyros Theodoropoulos
stheodoropoulos@mail.ntua.gr

Patrik Zajec
patrik.zajec@ijs.si

Jože M. Rožanec
joze.rozanec@ijs.si

Dimosthenis Kyriazis
dimos@unipi.gr

Panayiotis Tsanakas
panag@cs.ntua.gr

[1] School of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece

[2] Department of Digital Systems, University of Piraeus, Piraeus, Greece

[3] Jožef Stefan International Postgraduate School, Ljubljana, Slovenia