MOCCA: Multi-layer One-Class ClassificAtion for Anomaly Detection

Fabio Valerio Massoli📷, Fabrizio Falchi📷, Alperen Kantarci📷, Şeymanur Akti📷, Hazim Kemal Ekenel📷, Giuseppe Amato📷

Abstract—Anomalies are ubiquitous in all scientific fields and can express an unexpected event due to incomplete knowledge about the data distribution or an unknown process that suddenly comes into play and distorts the observations. Usually, due to such events' rarity, to train deep learning models on the Anomaly Detection (AD) task, scientists only rely on "normal" data, i.e., non-anomalous samples. Thus, letting the neural network infer the distribution beneath the input data. In such a context, we propose a novel framework, named Multi-layer One-Class ClassificAtion (MOCCA), to train and test deep learning models on the AD task. Specifically, we applied our approach to autoencoders. A key novelty in our work stems from the explicit optimization of the intermediate representations for the task at hand. Indeed, differently from commonly used approaches that consider a neural network as a single computational block, i.e., using the output of the last layer only, MOCCA explicitly leverages the multi-layer structure of deep architectures. Each layer's feature space is optimized for AD during training, while in the test phase, the deep representations extracted from the trained layers are combined to detect anomalies. With MOCCA, we split the training process into two steps. First, the autoencoder is trained on the reconstruction task only. Then, we only retain the encoder tasked with minimizing the L_2 distance between the output representation and a reference point, the anomaly-free training data centroid, at each considered layer. Subsequently, we combine the deep features extracted at the various trained layers of the encoder model to detect anomalies at inference time. To assess the performance of the models trained with MOCCA, we conduct extensive experiments on publicly available datasets, namely CIFAR10, MVTec AD, and ShanghaiTech. We show that our proposed method reaches comparable or superior performance to state-of-the-art approaches available in the literature. Finally, we provide a model analysis to give insights regarding the benefits of our training procedure.

Index Terms—Anomaly Detection, One-Class Classification, Deep Learning

I. INTRODUCTION

ANOMALIES represent a controversial phenomenon in the scientific world. Although they can lead to fascinating discoveries, sometimes they are a symptom of something unexpected that just happened. Even though they can manifest in different ways, all kinds of anomalies origin from a common basic principle: an unexpected prediction from a given theory from what is believed to be a proper answer.

Fabio Valerio Massoli, Fabrizio Falchi, and Giuseppe Amato are with the Institute of Information Science and Technologies (ISTI) - CNR, Pisa, 56124 Italy (e-mail: {fabio.massoli; fabrizio.falchi; giuseppe.amato}@isti.cnr.it).

Hazim Kemal Ekenel, Alperen Kantarci, and Şeymanur Akti are with the Department of Computer Engineering, Istanbul Technical University, Istanbul, Turkey (e-mail: {ekenel; kantarcia; akti15}@itu.edu.tr)
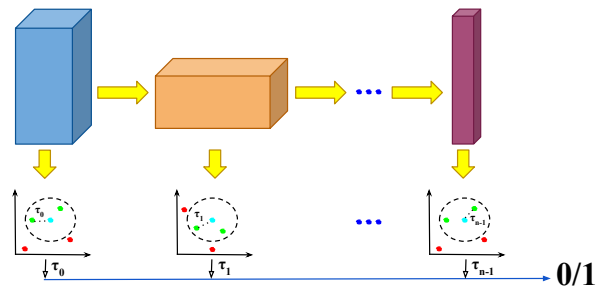
Fig. 1. Schematic representation of the MOCCA approach. Each feature space is represented as an $x - y$ plane. The cyan dots represent the centroids of the anomaly-free images while the green (red) dots represent the normal (anomalous) samples, respectively. τ_i is the distance between the deep representation of a given input image from the centroid at layer i.

Concerning the Deep Learning (DL) field, an anomaly might be thought of as an out-of-distribution sample presented as input to a Deep Neural Network (DNN). More specifically, from a statistical point of view [1, 2], we can discern among outliers and novelties that are described by the same probability distribution of the normal data and anomalies that are instead characterized by completely different statistics. Being able to detect such events is an attractive feature, especially concerning applications such as surveillance systems [3–6], medical diagnosis [7–10], fraud detection [11–14], and defect detection [15, 16]. Indeed the task of Anomaly Detection (AD) [17, 18] is among the most active research fields in the machine learning community.

Since the cost to collect large amounts of anomalous samples is prohibitive, the AD is usually considered as an unsupervised problem with the training databases containing non-anomalous class instances only. Thus, to detect anomalies, deep models are typically trained on in-manifold samples only to learn an effective boundary that captures the concept of normality from the distribution of one kind of data only. In recent years, One-Class (OC) approaches to AD have drawn the scientific community's interest. Especially, autoencoders [19–21] and GANs [22–24] based approaches reached the highest performance available in the literature.

In the Machine Learning (ML) field, commonly adopted approaches leverage the models' final output only, thus interpreting a neural network as a single computational block that performs an input-to-output mapping. Concerning such a point of view, throughout this manuscript, we refer to such an approach as "holistic" interpretation. Specifically, what we mean by "holistic" is that both the training and test phases rely

on the output of the last layer only, i.e., there is no information extracted from the intermediate levels of the architecture.

In such a context, our contribution stems from a different interpretation of the mapping represented by a DNN. We show that by leveraging the deep representations extracted at various depths in both the training and inference phases of a learning model, a neural network reaches higher performance on the AD task than when only the last layer's output is considered. We propose a novel framework, named Multi-layer One-Class ClassificAtion (MOCCA), to train and test deep learning models on the AD task. The innovation in our work is the explicit optimization of the intermediate representations and their use in the test phase for the task at hand. MOCCA leverages the multi-layer structure of deep architectures, differently from commonly used approaches that consider a neural network as a single computational block, i.e., using the output of the last layer only. During training, each layer's feature space is optimized for AD, whereas in the test phase, the deep representations extracted from the trained layers are combined to detect anomalies. To prove the effectiveness of our strategy, we apply it to autoencoders. Specifically, with MOCCA, we split the training process into two steps. First, the autoencoder is trained on the reconstruction task only. Then, we only retain the encoder tasked with minimizing the L_2 distance between the output representation and a reference point, the anomaly-free training data centroid, at each considered layer. Subsequently, we combine the deep features extracted at the various trained layers of the encoder model to detect anomalies at inference time. We show a schematic view of our approach in Figure 1. Our contributions can be summarized as follows:

- we formulate a "multi-layer" based approach to AD, named MOCCA, that explicitly optimizes the representations extracted at different layers of a deep learning model during training, and then combines them in the test phase to detect anomalies;
- we perform extensive experiments on publicly available single-image AD datasets, namely, CIFAR10 and MVTec AD [25], and empirically show that models trained with the MOCCA approach reach higher performance compared to the state-of-the-art;
- we perform experiments on the ShanghaiTech [26] dataset, and show that, even though our method is not tailored for video-based AD, it delivers models with performance comparable to state-of-the-art approaches specially designed for such a task. Thus, showing the high generalization capability of our technique;
- we perform a model analysis to give insights into how our approach works and empirically analyze the benefits of exploiting the representations generated at different layers of a learning model.

The remainder of the paper is organized as follows. In section II, we briefly review the related works, while in section III, we describe our approach to the anomaly detection task. In section IV and section V, we present the datasets we used and report the obtained results on them, respectively. In section VI we perform an analysis of the models, and, finally, in section VII, we conclude the paper.

II. RELATED WORKS

The latest approaches to the AD task are mainly based on reconstruction and discrimination techniques. Autoencoders [19, 27–29] and GANs [30–32] belong to the former class while the latter approach gathers techniques such as the one-class classification [33–35].

Concerning GAN-based approaches, in [32], the authors exploit a reconstruction technique that leverages an autoencoder and a CNN that are adversarially trained. In AnoGAN [10] the generator learns to reconstruct the input sample through latent space optimization, and the discriminator generates deep representations for both the original and the reconstructed samples, while in [36], the authors propose to learn an encoder network that maps the input samples directly to the generator's latent space. A slightly different approach is proposed in [23], where an explicit latent space minimization is obtained by learning an encoder model. The OC-GAN approach is introduced in [37], where authors use a denoising autoencoder network and a classifier in order to learn the latent representations of the normal samples in an adversarial manner.

In [38] variational autoencoders are used to detect anomalies by exploiting the reconstruction probability as the objective. In [39] the authors combine a reconstruction approach based on autoencoders with an autoregressive model that learns a factorization of the latent space distribution. In [40], the authors use the structural similarity index metric (SSIM) to train autoencoders while [41] propose the Inverse-Transform AutoEncoder (ITAE) based on the use of autoencoders that reconstruct images after the application of a set of specific transformations.

The One-Class (OC) approach has a long history starting from the study of shallow models. Indeed, first attempts in such a direction date back to the 2000s with the proposal of the One-Class SVM [42, 43]. In [44], a hybrid approach is proposed based on deep autoencoders and OC-SVM, while in [45] the authors trained their models with an OC-SVM equivalent loss function. One of the first proposals concerning an end-to-end training approach to OC-AD is proposed in [46], where the code generated by an encoder is mapped to a point within a hypersphere so that the normal samples remained inside of it while anomalous ones lay outside. Lastly, in [47], the authors use an encoder for getting the latent representations of the normal samples, and a pseudo-negative class is created using zero-centered Gaussian noise in the same latent space.

Most recently, Venkataramanan et al. [48] exploit a variational autoencoder combined with a specialized attention mechanism with the final goal of performing anomaly localization. In [49], the authors tackle the problem of the stability training of GANs when there are not lots of data available. A semi-supervised approach is proposed in [50], and in [51], the authors exploit a student–teacher framework to perform anomaly detection and pixel-precise anomaly segmentation at the same time. In [52], they leverage a multiple instance learning approach while in [20] a technique named MemAE is introduced where authors have added a memory module to deep autoencoders.

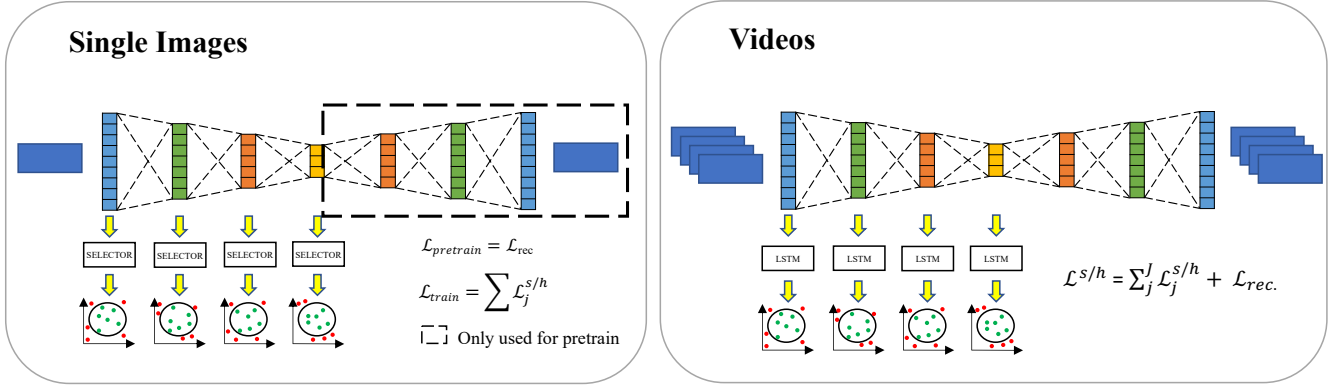Compared to all the works mentioned above, MOCCA

Fig. 2. Schematic representation of the MOCCA training. Left: two-stage training for single image input. Right: end-to-end training for video-based AD. To exploit the time correlation among frames, LSTMs are used instead of "selector" modules. The superscript s (h) refers to the *soft* (*hard*) boundary settings.

differs from them on two key aspects. On the one side, it exploits the deep representation extracted at various layers of the learning model, both at training and inference time, which contrasts to classical methodology in which only the final output is considered to fulfill the task. On the other hand, it does not make any assumption on the deep features' statistical distributions. Combining these two properties allows the model to adjust each single feature space at its best to accomplish the AD task.

III. PROPOSED APPROACH

As a general conception, DNNs are a sequence of transformations that approximate a function $f_\theta : \mathcal{X} \to \mathcal{Y}$ where $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}^m$ are the input and output space, respectively, and θ are the parameters to be learned at training time. We refer to such an approach as "holistic" (see section I for more details) in the sense that the entire net is considered as a single computational block that given an input, returns an output. As opposed to such a point of view, with MOCCA we adopt a "multi-layer" interpretation of the learning models where we consider a DNN as a sequence of single transformations each mapping its input to a more representative space:

$$f_\theta(\mathbf{x}) = \phi_m(\theta_m; \mathbf{o}_{m-1}) \circ \phi_{m-1}(\theta_{m-1}; \mathbf{o}_{m-2}) \circ .... \circ \phi_1(\theta_1; \mathbf{x})$$
(1)

where each ϕ_i term represents the operation performed by a specific layer, and the matrices θ_i represent their weights and biases. The output of each operation is reported as $\mathbf{o}_i$, while $\mathbf{x}$ is the network input.

Our intuition is that the outputs $\mathbf{o}_i$ of the various layers, i.e., the representations generated at different depths of a DNN, can be exploited to enhance the performance of a learning model on the AD task compared to when the entire decision process leverages the last layer output only. Indeed, it has been already shown in literature [53–55] that deep features extracted at various layers of a model can help a DNN to fulfill its task. However, it is not enough to combine the representations at

test time only. Instead, all the layers must be trained to a common aim.

As mentioned in section I, our base network is an autoencoder where both the encoder and the decoder are Deep Convolutional Neural Network (DCNN). With MOCCA we formulate the training process as a two-stage procedure in which we first train the full autoencoder on the reconstruction task only, and then we specialize only the encoder to detect anomalies by exploiting an OC-like objective [46] applied to different layers of the network. However, we empirically observe that a single-step end-to-end training, in which we optimize the reconstruction and the OC objectives simultaneously, is more effective than the two-step one for video-based AD. A schematic representation of the MOCCA training procedures is presented in Figure 2. As one can see from the figure, we process the model inner layers' output using "selector" and "LSTM" modules concerning single-image and video-based data type, respectively. Concerning the "selector" blocks, they are made of an average pooling operation or a two-layer neural network concerning the CIFAR10 and MVTec AD [25], respectively. Specifically, concerning the CIFAR10 dataset, we use only the pooling operation to fully assess the real advantages brought by MOCCA.

As mentioned above, we exploit the OC objective and we evaluate it by using the deep features extracted at different depths of the encoder model. Specifically, we considered two variants for such an objective function termed *soft-* and *hard-*boundary. The first one is expressed as follows:

$$\mathcal{L}_j^s = R_j^2 + \frac{1}{|B| \cdot \nu} \sum_i^{|B|} \max\{0, \| \phi_j(\mathbf{x}_i; \theta) - \mathbf{c}_j \|^2 - R_j^2\}$$
(2)

The goal of such a loss is to minimize the volume of the hypersphere at each layer j, centered at $\mathbf{c_j}$ and with radius R_j, that is interpreted as the boundary region for normal data [43]. Then, the goal of Equation 2 is to minimize the radius, R_j, of such spheres (one for each trained layer). In other words, we expect the "normal" data to lie within a sphere, at each layer, while the anomalous samples are expected to remain outside of

it. The second addend in the equation penalizes "normal" data points that lie outside the sphere after being passed through the network. The radius R_j is a scalar quantity evaluated as the $1 - \nu$ quantile of the features' distance distribution, in a mini-batch, from the centroid $\mathbf{c_j}$. We re-evaluate the radius at each layer at regular intervals while training. A decreasing value of the radius at each layer is an indicator of converging training. The other terms in the equation have the following interpretation: $|B|$ is the mini-batch size, ν is a hyperparameter that allows controlling the fraction of allowed outliers, ϕ_j represents the function that the layer j carries out, and $\mathbf{x}_i$ is the model input.

Concerning the *hard*-boundary loss, it is expressed as follows:

$$\mathcal{L}_j^h = \frac{1}{|B| \cdot \nu} \sum_{i}^{|B|} \parallel \phi_j(\mathbf{x}_i; \theta) - \mathbf{c}_j \parallel^2 \qquad (3)$$

Differently from Equation 2, Equation 3 simply tries to reduce as much as possible the distance of each sample from the layer's centroid by employing a quadratic loss.

After the first training step in which we tasked the full autoencoder with the reconstruction objective, we retain only the encoder and perform an initial forward step on the whole training dataset (that contains non-anomalous samples only) to extract deep features at different depths. Subsequently, we evaluate the centroids, at each layer, as the average of those features. We performed experiments in which we tested the hypothesis of using medoids instead of centroids, but we did not observe any improvement. Once we evaluate the centroids, they are kept fix while training the encoder. We also experimented with several strategies to re-evaluate them after a specific number of training iterations, but we did not observe tangible improvements. Regarding the video-based AD, we initialize the centroids at the beginning of the training, i.e., with the model not trained.

Considering a set of layers $\mathcal{J} = \{j \mid j = 0, 1, ...J\}$, we formalize the MOCCA objective, during the second-step of the training, as:

$$\mathcal{L}^{s/h} = \frac{1}{|\mathcal{J}|} \sum_{j}^{|J|} \mathcal{L}_j^{s/h} + \frac{\lambda}{2} \sum_{p}^{|P|} \parallel \theta_p \parallel^2 \qquad (4)$$

where $|J|$ is the number of layers we consider, and the sum runs over the layer indexes j. The last term of the objective is the L_2 regularization for the model parameters.

IV. DATASETS AND TRAINING

This section reports the used datasets and provides details about the training procedure that we adopt.

A. CIFAR10

The CIFAR10 dataset contains 50K training images and 10K test ones shared among ten different classes. We preprocess the images by applying a global contrast normalization procedure using the L_1 norm, and then we normalize them

to be in the range $[0, +1]$. Given each class, which we refer to as the "normal class", we have 5000 images to train the model, and we evaluate each model's performance on the whole test set. With such a training approach, the model only sees instances from the "normal class" and never sees any anomaly while learning.

B. MVTec

The MVTec AD [25] dataset comprises ∼3.6K and ∼1.7K high-resolution images to train and test DNNs, respectively, shared among 15 classes which are divided into two categories: textures (5 classes) and objects (10 classes). The dataset is split into two sets: one for training purposes containing "normal" images only and one specifically designed to test the models' performance. Specifically, the latter one contains anomalous images, with defects of different types and non-anomalous ones. We apply two different preprocessing operations to objects- and texture-type classes. Concerning the formers, we first resize the image to 128x128 pixels and then apply a random rotation in the range $[-\pi/4, +\pi/4]$ when the anomaly of the object is not related to its orientation. Instead, we first resize images to 512x512 pixels in the latter type of classes, and then we crop 64x64 non-overlapping patches used as input to the network. Moreover, we augment the data by exploiting a random rotation in the range $[0, +\pi/4]$. Finally, we normalize all the objects- and texture-type images to be in the range $[-1, +1]$. In Figure 3 we show an example of textures- and objects-type images from the dataset.

Fig. 3. Samples from different classes of the MVTec AD [25] dataset. Top: texture classes. Bottom: object classes. We highlight in red the anomalies.

C. ShanghaiTech

The ShanghaiTech [26] dataset is one of the largest video anomaly datasets. It comprises over 270,000 training frames from 13 scenes with complex light conditions and camera angles, accounting for 130 abnormal events. We follow the same preprocessing strategy as in [39], i.e., we use a MOG-based approach to estimate the background and remove it from the frames. By employing such a procedure, we eliminate the necessity of background estimation and let the model focus on foreground objects only. Given a video, we construct clips made by 16 frames to be used as input to the learning models. To exploit the temporal correlation among frames, we employ LSTM cells (we refer the reader to section III for more details about our models' architecture). Finally, we resize each frame to 256x512 pixels to feed models. In Figure 4 we report an example of "normal" and anomalous frames from two different videos.

Fig. 4. Samples of "normal" (left) and anomalous (right) frames from the ShanghaiTech [26] dataset. We highlight in red the anomalies.

D. Training details

Concerning the CIFAR10 dataset, we use a LeNet-like architecture as in [46], made of three convolutional layers and one fully connected layer after them. We use the Adam [56] optimizer for both pre-train the full architecture and train the encoder with learning rates of $1.e^{-3}$ and $1.e^{-4}$, respectively. We set the encoder code's size equals to 128 and the value of the parameter ν in the range [0, 0.1]. Finally, we use a batch size of 256. As we mentioned in section III, concerning the CIFAR10 dataset, we use an average pooling operation as the "selector" module. Thus, we emphasize that the higher performance reached by using MOCCA is not due to larger, deeper, or more models. Instead, the benefits of using MOCCA stand from its ability to exploit the representations generated at different depths of a learning model. To our aim, we train ten different seeded models on each class, considering the other nine as anomalies. Such a procedure allows us to estimate the mean response of our approach and its standard deviation.

Regarding the MVTec AD [25] and the ShanghaiTech [26] datasets, we use a residual-like structure that comprises four and five residual blocks, respectively, followed by two fully connected layers. For this dataset, the "selector" blocks consist of a convolutional layer followed by a pooling operation, a batch norm layer, and a final fully connected layer. In video-based AD, we substitute the "selector" networks with LSTM cells to exploit the time correlation among the frames within a given input clip. To train models on those two datasets, we use again the Adam [56] optimizer and a learning rate in the set $\{10^{-2}, 10^{-3}\}$ that we drop by a factor of ten at specific epochs depending on the class under study. Being each class of each dataset an independent AD problem, we use different hyperparameters to train the models on each of them. Moreover, we do not always use the same set of layers to evaluate the objective in Equation 4. Indeed, we train the models by using different layer combinations and finally select the best performing one in each class.

To allow the researchers to reproduce our work, we made the code publicly available on GitHub[1].

V. EXPERIMENTS

In this section, we report our experimental results. However, before that, we describe the various metrics we use to assess the models' performance.

[1] https://github.com/fvmassoli/mocca-anomaly-detection.git

A. Metrics

To assess the performance of the models trained with MOCCA and compare them to the other approaches in the literature, we exploit two metrics: the Area Under the Curve (AUC) and the maximum Balanced Accuracy (maxBA). The former metric is the area under the Receiver Operating Characteristics curve. Instead, concerning the latter, the Balanced Accuracy (BA) represents the arithmetic mean between the sensitivity, i.e., percentage of anomalous samples correctly detected, and the specificity, i.e., same as the sensitivity but for non-anomalous samples:

$$\text{BA} = \frac{TP}{2 \cdot (TP + FN)} + \frac{TN}{2 \cdot (TN + FP)} \quad (5)$$

where TP and FN are the true positives and the false negatives, respectively, and TN and FP are the true negatives and the false positives, respectively.

In the AD context, it is useful to quote both the AUC and the maxBA metrics. The former one provides an aggregate measure of the performance of a model across all possible classification thresholds. Instead, maxBA is a measure of performance at a specific threshold that could be used in production. It selects the threshold for which the balanced accuracy measure, i.e., the average among the correctly classified images for anomalous (true positives) and anomaly-free test images (true negatives), is maximum and reports the obtained BA. We evaluate both metrics only on the MVTec AD [25] dataset since for the CIFAR10 and ShanghaiTech [26] datasets we only found the AUC values reported in the literature. Concerning the anomaly score for a given input image, we evaluate its value as:

$$\tau_j(\mathbf{x}) = \| \phi_j(\mathbf{x}, \theta) - \mathbf{c}_j \|^2 \quad (6)$$

$$\gamma(\mathbf{x}) = \frac{1}{|\mathcal{J}|} \sum_j^{|\mathcal{J}|} \begin{cases} \tau_j(\mathbf{x}) & \text{hard boundary} \\ \tau_j(\mathbf{x}) - R_j^2 & \text{soft boundary} \end{cases}$$

where $\mathbf{x}$ is the input image, $\mathcal{J} = \{j \mid j = 0, 1, ...J\}$ is the set of layers we consider, $\phi_j(\mathbf{x}, \theta)$ is the feature vector extracted at layer j, and $\mathbf{c}_j$ and R_j are the center of the hypersphere and its radius at the layer j, respectively and γ is the anomaly score. We refer the reader to section III for further details on the meaning of the boundaries. Concerning the textures-type classes from the MVTec AD [25], we evaluate the anomaly score as the maximum among the scores relative to each of the 64x64 patches of the given image:

$$\gamma^{h/s}(\mathbf{x}) = \max\{\gamma^{h/s}(\text{patch}_i)) \mid i = 1, 2, ..., 64\} \quad (7)$$

where the superscripts s and h correspond to when we apply a "soft" or "hard" boundary while training the model, respectively. More details on how we extract patches from a single image can be found in subsection IV-B. Lastly, considering video-based input we consider a single input clip as made of 16 frames. We then apply a sliding window technique to move through all the frames of a given video and

Class	VAE [57]⋆	Pix CNN [58]⋆	DCAE†	AnoGAN [10]†	LSA [39]	Deep SVDD$_{(s)}$ [46]	MOCCA$_{(s)}$	Deep SVDD$_{(h)}$ [46]	MOCCA$_{(h)}$
0	0.688	0.788	$0.601 \pm .007$	$0.671 \pm .025$	**0.735**	$0.617 \pm .042$	$0.626 \pm .021$	$0.617 \pm .041$	$0.660 \pm .015$
1	0.403	0.428	$0.574 \pm .029$	$0.547 \pm .034$	0.580	$0.648 \pm .014$	$\underline{\mathbf{0.746}} \pm \mathbf{.008}$	$0.659 \pm .021$	$\mathbf{0.705} \pm \mathbf{.013}$
2	0.679	0.617	$0.489 \pm .024$	$0.529 \pm .030$	**0.690**	$0.495 \pm .014$	$0.575 \pm .018$	$0.508 \pm .008$	$0.524 \pm .010$
3	0.528	0.574	$0.584 \pm .012$	$0.545 \pm .019$	0.542	$0.560 \pm .011$	$0.578 \pm .011$	$0.591 \pm .014$	$\mathbf{0.601} \pm \mathbf{.006}$
4	0.748	0.511	$0.540 \pm .013$	$0.651 \pm .032$	**0.761**	$0.599 \pm .011$	$0.615 \pm .012$	$0.609 \pm .011$	$0.609 \pm .012$
5	0.519	0.571	$0.622 \pm .018$	$0.603 \pm .018$	0.546	$0.621 \pm .024$	$\mathbf{0.663} \pm \mathbf{.010}$	$0.657 \pm .025$	$\underline{\mathbf{0.684}} \pm \mathbf{.016}$
6	0.695	0.422	$0.512 \pm .052$	$0.585 \pm .014$	**0.751**	$0.678 \pm .024$	$0.674 \pm .012$	$0.677 \pm .026$	$0.671 \pm .005$
7	0.500	0.454	$0.586 \pm .029$	$0.625 \pm .008$	0.535	$0.652 \pm .010$	$\underline{\mathbf{0.721}} \pm \mathbf{.004}$	$0.673 \pm .009$	$\mathbf{0.685} \pm \mathbf{.010}$
8	0.700	0.715	$0.768 \pm .014$	$0.758 \pm .041$	0.717	$0.756 \pm .017$	$\mathbf{0.791} \pm \mathbf{.012}$	$0.759 \pm .012$	$\underline{\mathbf{0.792}} \pm \mathbf{.008}$
9	0.398	0.426	$0.673 \pm .030$	$0.665 \pm .028$	0.548	$0.710 \pm .011$	$\underline{\mathbf{0.773}} \pm \mathbf{.010}$	$0.731 \pm .012$	$\mathbf{0.758} \pm \mathbf{.007}$

⋆Values reported in [39]; †Values reported in [46]

TABLE I

AUC FOR THE CIFAR10 DATASET. THE SUBSCRIPTS (s) AND (h) REFER TO THE *soft* AND *hard* BOUNDARIES, RESPECTIVELY. WE EMPHASIZE IN BOLD THE PERFORMANCE OF THE BEST MODELS. WHENEVER OUR MODELS OVERCOME THE SotA WITH BOTH THE TYPE OF BOUNDARIES, WE UNDERLINE THE BEST OF THE TWO. WE ONLY REPORT ERRORS FROM OTHERS WHEN AVAILABLE IN THE REFERENCE PAPER.

construct the input clips. Since each frame can appear multiple times across different clips, we evaluate its score as the mean value among all of its scores. Moreover, a single frame can have different scores in different clips having a different time correlation, captured by the LSTMs (see Figure 2), with all the other frames. For such a reason, we normalize the score of each frame to the maximum and minimum values of the scores within the clips in which the frame under analysis is present:

$$\gamma^{h/s}(\mathbf{x}_i) = \frac{\langle\gamma^{h/s}(\mathbf{x}_i)\rangle - \max_{\text{clips}}\langle\gamma^{h/s}(\mathbf{x}_i)\rangle}{\max_{\text{clips}}\langle\gamma^{h/s}(\mathbf{x}_i)\rangle - \min_{\text{clips}}\langle\gamma^{h/s}(\mathbf{x}_i)\rangle} \quad (8)$$

Finally, we add a reconstruction term to the score.

B. Experimental results - CIFAR10

Concerning the CIFAR10 dataset, we instantiate each class as a single AD problem, and we train ten different seeded models on each of them. Such a procedure allows us to quote a mean AUC value and the corresponding variance. We report the results in Table I.

As we can see from Table I, our approach reaches the highest performance on six out of ten classes. Moreover, on class-1, class-5, class-7, class-8, and class-9, the MOCCA method performs better than the state-of-the-art (SotA) results concerning both the "soft" and the "hard" boundaries. As reported in subsection IV-D, on the CIFAR10 dataset we use a LeNet-like architecture as in [46]. Moreover, to better emphasize that our approach's higher performance is not due to a mere addition of more models to the baseline, we use averaging pooling layers as "selectors" blocks. Thus, since we use the same architecture as in [46], we can conclude that the higher performance of our models are only due to the use of MOCCA and not because we use deeper models or because we add more branches to the base architecture. To summarize the previous results, we report in Table II the AUC values, for each model in Table I, averaged among all the ten classes of the dataset.

From Table II, it is clear that our approach reaches the highest performance concerning both types of boundary settings. Moreover, we can appreciate that we obtain higher performance, also considering larger models such as LSA [39].

	Average AUC
VAE [57]⋆	$0.586 \pm .039$
Pix CNN [58]⋆	$0.551 \pm .038$
DCAE†	$0.595 \pm .024$
AnoGAN [10]†	$0.618 \pm .021$
LSA [39]	$0.640 \pm .029$
Deep SVDD$_{(s)}$ [46]	$0.634 \pm .022$
Deep SVDD$_{(h)}$ [46]	$0.648 \pm .022$
MOCCA$_{(s)}$	$\underline{\mathbf{0.676}} \pm \mathbf{.024}$
MOCCA$_{(h)}$	$\mathbf{0.669} \pm \mathbf{.023}$

⋆Values reported in [39]; †Values reported in [46]

TABLE II

AUC AVERAGED AMONG ALL CLASSES OF THE CIFAR10 DATASET. THE SUBSCRIPTS (s) AND (h) REFER TO THE *soft* AND *hard* BOUNDARIES, RESPECTIVELY. WE EMPHASIZE IN BOLD THE PERFORMANCE OF THE BEST MODELS. WHENEVER OUR MODELS OVERCOME THE SotA WITH BOTH THE TYPE OF BOUNDARIES, WE UNDERLINE THE BEST OF THE TWO.

C. Experimental results - MVTec AD

Regarding the MVTec AD [25] dataset, also, in this case, we consider each class as an independent AD problem. As reported in subsection IV-B, the dataset classes are divided into texture- and object-like sets. For each class, we report the maxBA and the AUC in Table III and Table IV, respectively. Regarding the texture-type of classes, we see from the tables that the MOCCA approach allows our models to reach the highest performance on three out of five classes concerning both the *hard* and *soft* boundaries. Similar reasonings hold in the case of object-type classes, too. Concerning the results from [48], it is important to highlight that, even though we report their results, they should not directly compared with others. The reason for that is because in [48], the models are trained on more data rather than on MVTec AD only. Thus, those results are not directly comparable with the other methods. Due to the very low number of test images available in the dataset, typically large variations in the performance of the model are observed among the different classes. Thus, to better compare the performance of the various approaches, we report in Table V the overall mean values for the maxBA and the AUC evaluated among all classes of the dataset.

From the table, we conclude that the MOCCA approach allows us to reach the highest performance on both types of metrics considering both the *soft* and *hard* type of boundary.

	Textures					Objects									
	Carpet	Grid	Leather	Tile	Wood	Bottle	Cable	Capsule	Hazelnut	MetalNut	Pill	Screw	Toothbrush	Transistor	Zipper
AE_{SSIM} [40]*	0.67	0.69	0.46	0.52	0.83	0.88	0.61	0.61	0.54	0.54	0.60	0.51	0.74	0.52	0.80
AE_{L2} [40]*	0.50	0.78	0.44	0.77	0.74	0.80	0.56	0.62	**0.88**	0.73	0.62	0.69	**0.98**	0.71	0.80
AnoGAN [10]†	0.49	0.51	0.52	0.51	0.68	0.69	0.53	0.58	0.50	0.50	0.62	0.35	0.57	0.67	0.59
VAE-grad [59]†	0.67	0.83	0.71	0.81	0.89	0.86	0.56	**0.86**	0.74	0.78	0.80	0.71	0.89	0.70	0.67
AVID [60]†	0.70	0.59	0.58	0.66	0.83	0.88	0.64	0.85	0.86	0.63	**0.86**	0.66	0.73	0.58	**0.84**
EGBAD [36]‡	0.60	0.50	0.65	0.73	0.80	0.68	0.66	0.55	0.50	0.55	0.63	0.50	0.48	0.68	0.59
CBiGAN [61]	0.60	**0.99**	0.87	**0.84**	0.88	0.84	**0.73**	0.58	0.75	0.67	0.76	0.67	0.97	0.74	0.55
$\text{MOCCA}_{(s)}$	<u>**0.81**</u>	0.85	<u>**0.96**</u>	0.80	<u>**0.97**</u>	**0.90**	0.72	0.77	0.77	<u>**0.85**</u>	0.81	<u>**0.82**</u>	0.93	<u>**0.77**</u>	0.78
$\text{MOCCA}_{(h)}$	**0.74**	0.76	**0.91**	0.78	**0.94**	**0.90**	0.68	0.75	0.76	**0.80**	0.69	**0.80**	0.91	<u>**0.81**</u>	0.78

*Values reported in [25]; †Values reported in [62]; ‡Values reported in [61]

TABLE III

MAXBA FOR ALL THE CLASSES OF THE MVTEC AD [25] DATASET. THE SUBSCRIPTS (s) AND (h) REFER TO THE *soft* AND *hard* BOUNDARIES, RESPECTIVELY. WE EMPHASIZE IN BOLD THE PERFORMANCE OF THE BEST MODELS. WHENEVER OUR MODELS OVERCOME THE SOTA WITH BOTH THE TYPE OF BOUNDARIES, WE UNDERLINE THE BEST OF THE TWO.

	Textures					Objects									
	Carpet	Grid	Leather	Tile	Wood	Bottle	Cable	Capsule	Hazelnut	MetalNut	Pill	Screw	Toothbrush	Transistor	Zipper
AE_{L2} [40]†	0.64	0.83	0.80	0.74	0.97	0.65	0.64	0.62	0.73	0.64	0.77	**1.00**	0.77	0.65	0.87
GeoTrans [63]†	0.44	0.62	0.84	0.42	0.61	0.74	0.78	0.67	0.36	0.81	0.63	0.50	0.97	0.87	0.82
GANomaly [23]†	0.70	0.71	0.84	0.79	0.83	0.89	0.76	0.73	0.79	0.70	0.74	0.75	0.65	0.79	0.75
ITAE [41]	0.71	0.88	0.86	0.74	0.92	0.94	**0.83**	0.68	**0.86**	0.67	0.79	**1.00**	**1.00**	0.84	**0.88**
EGBAD [36]*	0.52	0.54	0.55	0.79	0.91	0.63	0.68	0.52	0.43	0.47	0.57	0.46	0.64	0.73	0.58
CBiGAN [61]	0.55	**0.99**	0.83	**0.91**	0.95	0.87	0.81	0.56	0.77	0.63	0.81	0.58	0.94	0.77	0.53
CAVGA-R_u [48]**	*0.73*	*0.75*	*0.71*	*0.70*	*0.85*	*0.89*	*0.63*	*0.83*	*0.84*	*0.67*	*0.88*	*0.77*	*0.91*	*0.73*	*0.87*
CAVGA-D_u [48]**	*0.78*	*0.78*	*0.75*	*0.72*	*0.88*	*0.91*	*0.67*	*0.87*	*0.87*	*0.71*	*0.91*	*0.78*	*0.97*	*0.75*	*0.94*
$\text{MOCCA}_{(s)}$	<u>**0.86**</u>	0.87	<u>**0.98**</u>	0.89	<u>**1.00**</u>	**0.95**	0.76	<u>**0.82**</u>	0.80	<u>**0.85**</u>	0.82	0.84	0.97	**0.88**	0.84
$\text{MOCCA}_{(h)}$	**0.74**	0.81	**0.95**	0.85	**0.97**	0.93	0.72	**0.79**	0.78	**0.84**	0.73	0.80	0.95	0.84	0.82

†Values reported in [41]; *Values reported in [61]; **Results obtained by using more data - should NOT be directly compared to all the other methods

TABLE IV

AUC FOR ALL THE CLASSES OF THE MVTEC AD [25] DATASET. THE SUBSCRIPTS (s) AND (h) REFER TO THE *soft* AND *hard* BOUNDARIES, RESPECTIVELY. WE EMPHASIZE IN BOLD THE PERFORMANCE OF THE BEST MODELS. WHENEVER OUR MODELS OVERCOME THE SOTA WITH BOTH THE TYPE OF BOUNDARIES, WE UNDERLINE THE BEST OF THE TWO.

	Overall Mean	
	maxBA	AUC
AE_{SSIM} [40]*	0.63	-
AE_{L2} [40]*	0.71	0.75
AnoGAN [10]†	0.55	-
VAE-grad [59]†	0.77	-
AVID [60]†	0.73	-
EGBAD [36]‡	0.61	0.60
GeoTrans [63]††	-	0.67
GANomaly [23]††	-	0.76
ITAE [41]††	-	0.84
CBiGAN [61]	0.76	0.77
$\text{MOCCA}_{(s)}$	<u>**0.83**</u>	0.88
$\text{MOCCA}_{(h)}$	**0.80**	0.83

*Values reported in [25]; †Values reported in [62]
‡Values reported in [61]; ††Values reported in [41]

TABLE V

AVERAGE MAXBA AND AUC FROM TABLE III AND TABLE IV. THE SUBSCRIPTS (s) AND (h) REFER TO THE *soft* AND *hard* BOUNDARIES, RESPECTIVELY. WE EMPHASIZE IN BOLD THE PERFORMANCE OF THE BEST MODELS. WHENEVER OUR MODELS OVERCOME THE SOTA WITH BOTH THE TYPE OF BOUNDARIES, WE UNDERLINE THE BEST OF THE TWO. THE "-" SYMBOL MEANS THAT THE AUTHORS DID NOT REPORT THE VALUE.

D. Experimental results - ShanghaiTech

Differently from the CIFAR10 and MVTec AD [25] datasets, the ShanghaiTech [26] concerns the video-based AD task. Although we test models trained with MOCCA against such a protocol, it is essential to stress that our approach is not specially designed for the video-based scenario. We report our results in Table VI and others available in the literature.

	AUC
AE-Conv2D [6]†	0.609
TSC [64]†	0.679
Stack RNN [64]†	0.680
AE-Conv3D [65]†	0.697
MemAE [20]†	0.712
LSA [39]	0.725
ITAE [41]	0.725
FFP+MC [66]	0.728
Mem-Guided (w/o Mem.) [67]	0.668
Mem-Guided (w/ Mem.) [67]	0.705
MemAE-nonSpar [20]	0.688
MemAE [20]	0.712
Clustering-Driven [68]	**0.733**
$\text{MOCCA}_{(s)}$	0.730
$\text{MOCCA}_{(h)}$	0.725

†Values reported in [41]

TABLE VI

AUC VALUES FOR THE SHANGHAITECH [26] DATASET. THE SUBSCRIPTS (s) AND (h) REFER TO THE *soft* AND *hard* BOUNDARIES, RESPECTIVELY. WE REPORT IN BOLD THE PERFORMANCE OF THE BEST MODEL.

From Table VI, we can see that our approach's performance

is utterly comparable to the current SotA models, specifically designed to handle video-based input. Thus, showing that our method is applicable to both the image- and video-based anomaly detection tasks. Indeed, the only modification we apply to MOCCA for video-based contexts is to move to a single-step training, based on the same objectives, and to substistute the "selector" modules with LSTMs.

VI. MODEL ANALYSIS

In this section, we look in more detail at the behavior of our models. First, we focus on an ablation study to show the impact of using a different number of layers to evaluate a specific image's anomaly score. Specifically, we prove that with MOCCA, we effectively succeed in exploiting the deep representations extracted at different depths of a DNN. To our aim, we perform the ablation study considering the "Leather" class of the MVTec AD [25] dataset. We report the results in Table VII.

Layer index	maxBA	
	hard boundary	soft boundary
6	0.819 ± .020	0.839 ± .010
5, 6	0.855 ± .021	0.840 ± .012
4, 5, 6	0.906 ± .001	**0.955 ± .007**
3, 4, 5, 6	**0.912 ± .002**	0.935 ± .005
2, 3, 4, 5, 6	0.903 ± .003	0.947 ± .005
1, 2, 3, 4, 5, 6	0.865 ± .004	0.948 ± .003
0, 1, 2, 3, 4, 5, 6	0.873 ± .002	0.924 ± .001

TABLE VII
ABLATION STUDY CONCERNING THE "LEATHER" CLASS OF THE MVTEC AD [25] DATASET. WE REPORT THE MAXBA FOR THE *hard* AND *soft* BOUNDARY SETTINGS. WE HIGHLIGHT IN BOLD THE BEST RESULTS.

As described in subsection IV-D, the encoder's architecture consists of four residual blocks followed by two fully connected layers. The indexes in the first column of Table VII correspond to the layers' ordering where the 0-th layer is the closest to the input. The results in Table VII should be interpreted as follows. Each row in the table represents a different model that we trained with MOCCA by considering the output from the layers listed in the first column. For example, the first row represents the results we obtained by considering the output (in the training and test phases) from the last layer only, while in the second row we consider the layer 5 and 6 together. We aim at showing that by exploiting the output at different layers while training a learning model, we can use the output from those same layers at inference time to enhance the network's discrimination power. On the contrary, we experimentally observed that training the model using the last layer's output only and then using more layers at inference time always gave worse results. Such an observation is one of the key points on which we base our approach.

As it is clear from the table, independently from the type of boundary we apply, we obtain higher results by utilizing more layers. This result supports our intuition that the features extracted at different depths help to detect anomalies in the input images. By carefully looking at Table VII we notice that the maxBA improves until we add layers 4 and 5 to the last one. Moreover, we can notice that, in the case of the

hard boundary setting, we can obtain a slight improvement by adding layer 3. Finally, we notice that by adding more layers, we do not see any further improvement. We can interpret such behavior by considering that since the first layers are closer to the input data, they specialize on simple patterns. On the contrary, higher layers generate representations that amplify aspects of the input that are important for discrimination [69], thus more useful to fulfill the final task. Hence, adding layers that are too close to the input data does not improve the learning model's overall performance.

We then focus our attention on the distribution of the distances among the features and the centroids, of a given class, at different layers. Specifically, we compare our training approach against the "holistic" approach, i.e., when the learning model is considered a single computational block. As specified in section I, "holistic" refers to the approach similar to [46] where the last layer's output only is used to train and test the encoder on the AD task. To our aim, we train two identical models once with the MOCCA approach and then by evaluating the OC loss on the last layer only ("holistic"). We report the resulting Cumulative Density Function (CDF) in Figure 5.
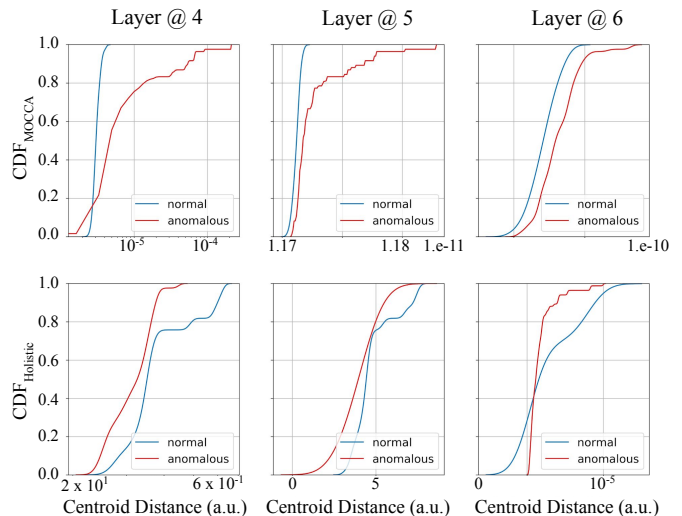

Fig. 5. CDF of the test images distance from the "normal" class centroid for the MOCCA approach (top row) and for the "holistic" one (bottom row). The blue (red) line represents the CDF of "normal" (anomalous) images.

Concerning the model trained with MOCCA, we see that the distributions for "normal" images always lies at the left of the corresponding for anomalous ones (as one would expect). Moreover, we see that the CDFs of "normal" images rise faster than the ones of anomalous samples. Thus, allowing one to set a more discriminative threshold on the anomaly score. On the contrary, we see that by considering the last layer only while training on the AD task, the distributions of distances for anomalous and "normal" images are highly overlapped even in the last layer. Thus, by training with MOCCA, we have a double gain: on the one side, we obtain discriminative deep features from more layers, and on the other hand, we are able to set more discriminative thresholds.

VII. Conclusions

The anomaly detection task is still an open challenge in many scientific fields. Several approaches have been proposed to tackle this problem in the context of deep learning, typically based on an unsupervised training paradigm. Indeed, being rare events, collecting anomalous samples to construct a supervised training dataset might be extremely expensive. Thus, approaches in which neural networks automatically learn the concept of "normality" from non-anomalous data only represent a promising solution.

We propose to adopt a multi-layer approach, named MOCCA, to exploit the output of a deep model at different depths to detect anomalous input in the one-class setting. Differently from the usual "holistic" interpretation of a learning model in which a neural network is considered a single computational block, MOCCA explicitly leverages the networks' multi-layer composition. Specifically, we show that such an approach enhances a neural network's discrimination capability. We conduct extensive experiments on three different datasets and perform an analysis of the models to support our intuitions. We test our method against the single-image AD task showing that it improves the state-of-the-art both on the CIFAR10 and MVTec AD datasets. Specifically, concerning the performance averaged among all the classes, MOCCA improves upon the literature results with both the *soft* and *hard* type of boundary. We acknowledge the best improvement concerning the overall maxBA on the MVTec AD dataset that overcomes the state-of-the-art results by 6%. Moreover, even though our approach is not tailored for the video-based AD task, we test it also using such a protocol by employing the ShanghaiTech dataset. From the experimental results, we see that with MOCCA, the models' performance is utterly comparable to what was obtained by approaches specially designed for such a task. Thus, showing the high generalization capability of our method.

Finally, we report insights about the behavior of models trained with MOCCA by performing an ablation study and reporting the different CDFs of the distance of the deep representations from the centroids of a given class across different layers. Such an analysis, pointed out that the benefits from using MOCCA are two-fold: on the one side, we obtain discriminative deep features from more layers, and on the other hand, we are able to set more discriminative thresholds.

References

[1] D. M. Hawkins, *Identification of outliers*. Springer, 1980, vol. 11. 1

[2] F. Y. Edgeworth, "Xli. on discordant observations," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 23, no. 143, pp. 364–375, 1887. 1

[3] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, 2019. 1

[4] S. Zavrak and M. İskefiyeli, "Anomaly-based intrusion detection from network flow features using variational autoencoder," *IEEE Access*, vol. 8, pp. 108 346–108 358, 2020. 1

[5] W. Alhakami, A. ALharbi, S. Bourouis, R. Alroobaea, and N. Bouguila, "Network anomaly intrusion detection using a non-parametric bayesian approach and feature selection," *IEEE Access*, vol. 7, 2019. 1

[6] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *CCVPR*, 2016. 1, 7

[7] R. G. Stafford, J. Beutel *et al.*, "Application of neural networks as an aid in medical diagnosis and general anomaly detection," Jul. 19 1994, US Patent 5,331,550. 1

[8] T. Fernando, S. Denman, D. Ahmedt-Aristizabal, S. Sridharan, K. R. Laurens, P. Johnston, and C. Fookes, "Neural memory plasticity for medical anomaly detection," *Neural Networks*, 2020. 1

[9] K. Ouardini, H. Yang, B. Unnikrishnan, M. Romain, C. Garcin, H. Zenati, J. P. Campbell, M. F. Chiang, J. Kalpathy-Cramer, V. Chandrasekhar *et al.*, "Towards practical unsupervised anomaly detection on retinal images," in *DART*. Springer, 2019, pp. 225–234. 1

[10] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *ICIPMI*, 2017. 1, 2, 6, 7

[11] A. Roy, J. Sun, R. Mahoney, L. Alonzi, S. Adams, and P. Beling, "Deep learning detecting fraud in credit card transactions," in *SIEDS*, 2018. 1

[12] A. Pumsirirat and L. Yan, "Credit card fraud detection using deep learning based on auto-encoder and restricted boltzmann machine," *IJACSA*, vol. 9, no. 1, pp. 18–25, 2018. 1

[13] B. Lebichot, Y.-A. Le Borgne, L. He-Guelton, F. Oblé, and G. Bontempi, "Deep-learning domain adaptation techniques for credit cards fraud detection," in *INNSBDDL*. Springer, 2019, pp. 78–88. 1

[14] U. Fiore, A. De Santis, F. Perla, P. Zanetti, and F. Palmieri, "Using generative adversarial networks for improving classification effectiveness in credit card fraud detection," *Information Sciences*, vol. 479, pp. 448–455, 2019. 1

[15] K. Tout, F. Retraint, and R. Cogranne, "Automatic vision system for wheel surface inspection and monitoring," in *ASNT Annual Conference 2017*, 2017, pp. 207–216. 1

[16] A. Kumar, "Computer-vision-based fabric defect detection: A survey," *IEEE TIE*, vol. 55, no. 1, pp. 348–363, 2008. 1

[17] C. C. Aggarwal, "Outlier analysis," in *Data mining*. Springer, 2015, pp. 237–263. 1

[18] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv:1901.03407*, 2019. 1

[19] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *ACM SIGKDD*, 2017, pp. 665–674. 1, 2

[20] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh, and A. v. d. Hengel, "Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection," in *Proc. ICCV*, 2019, pp. 1705–1714. 1, 2, 7

[21] X. Wang, Y. Du, S. Lin, P. Cui, Y. Shen, and Y. Yang, "adVAE: A self-adversarial variational autoencoder with Gaussian anomaly prior knowledge for anomaly detection," *KBS*, vol. 190, p. 105187, 2020. 1

[22] T. Schlegl, P. Seeböck, S. M. Waldstein, G. Langs, and U. Schmidt-Erfurth, "f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks," *MIA*, vol. 54, pp. 30–44, 2019. 1

[23] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "GANomaly: Semi-supervised anomaly detection via adversarial training," in *ACCV*. Springer, 2018, pp. 622–637. 1, 2, 7

[24] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks," in *ICANN*. Springer, 2019, pp. 703–716. 1

[25] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "MVTec AD–a comprehensive real-world dataset for unsupervised anomaly detection," in *Proc. CVPR*, 2019. 2, 3, 4, 5, 6, 7, 8

[26] W. Luo, W. Liu, and S. Gao, "A revisit of sparse coding based anomaly detection in stacked RNN framework," in *ICCV*, 2017. 2, 4, 5, 7

[27] A. Borghesi, A. Bartolini, M. Lombardi, M. Milano, and L. Benini, "A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems," *Engineering Applications of Artificial Intelligence*, vol. 85, pp. 634–644, 2019. 2

[28] N. Pawlowski, M. C. Lee, M. Rajchl, S. McDonagh, E. Ferrante, K. Kamnitsas, S. Cooke, S. Stevenson, A. Khetani, T. Newman *et al.*, "Unsupervised lesion detection in brain CT using bayesian convolutional autoencoders," in *MIDL*, 2018. 2

[29] Y. Li, X. Huang, J. Li, M. Du, and N. Zou, "SpecAE: Spectral autoencoder for anomaly detection in attributed networks," in *Proc. CIKM*, 2019, pp. 2233–2236. 2

[30] K. Zhou, S. Gao, J. Cheng, Z. Gu, H. Fu, Z. Tu, J. Yang, Y. Zhao, and J. Liu, "Sparse-GAN: Sparsity-constrained generative adversarial network for anomaly detection in retinal OCT image," in *ISBI*. IEEE, 2020, pp. 1227–1231. 2

[31] P. C. Ngo, A. A. Winarto, C. K. L. Kou, S. Park, F. Akram, and H. K. Lee, "Fence GAN: towards better anomaly detection," in *ICTAI*. IEEE, 2019, pp. 141–148. 2

[32] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli, "Adversarially learned one-class classifier for novelty detection," in *CVPR*, 2018. 2

[33] J. Wang and A. Cherian, "GODS: Generalized one-class discriminative subspaces for anomaly detection," in *Proc. ICCV*, 2019. 2

[34] E. Hong and Y. Choe, "Latent feature decentralization loss for one-class anomaly detection," *IEEE Access*, vol. 8, 2020. 2

[35] I. Razzak and T. M. Khan, "One-class support tensor machines with bounded hinge loss function for anomaly detection," in *2020 Int. Joint Conf. on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8. 2

[36] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar, "Efficient GAN-based anomaly detection," *arXiv:1802.06222*, 2018. 2, 7

[37] P. Perera, R. Nallapati, and B. Xiang, "OCGAN: One-class novelty detection using GANs with constrained latent representations," in *CVPR*, 2019. 2

[38] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lecture on IE*, vol. 2, pp. 1–18, 2015. 2

[39] D. Abati, A. Porrello, S. Calderara, and R. Cucchiara, "Latent space autoregression for novelty detection," in *CVPR*, 2019. 2, 4, 6, 7

[40] P. Bergmann, S. Löwe, M. Fauser, D. Sattlegger, and C. Steger, "Improving unsupervised defect segmentation by applying structural similarity to autoencoders," *arXiv:1807.02011*, 2018. 2, 7

[41] C. Huang, J. Cao, F. Ye, M. Li, Y. Zhang, and C. Lu, "Inverse-transform autoencoder for anomaly detection," *arXiv:1911.10676*, 2019. 2, 7

[42] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," *NeurIPS*, 2000. 2

[43] D. M. J. Tax and R. P. W. Ruin, "Support vector data description," *Machine Learning*, vol. 54, pp. 45–66, 2004. 2, 3

[44] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning," *PR*, vol. 58, pp. 121–134, 2016. 2

[45] R. Chalapathy, A. K. Menon, and S. Chawla, "Anomaly detection using one-class neural networks," *arXiv:1802.06360*, 2018. 2

[46] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Int. Conf. on Machine Learning*, 2018, pp. 4393–4402. 2, 3, 5, 6, 8

[47] P. Oza and V. M. Patel, "One class convolutional neural network," *IEEE Signal Processing Letters*, vol. 26, no. 2, pp. 277–281, 2019. 2

[48] S. Venkataramanan, K.-C. Peng, R. V. Singh, and A. Mahalanobis, "Attention guided anomaly localization in images," in *European Conference on Computer Vision*. Springer, 2020, pp. 485–503. 2, 6, 7

[49] M. Z. Zaheer, J.-h. Lee, M. Astrid, and S.-I. Lee, "Old is gold: Redefining the adversarially learned one-class classifier training paradigm," in *CVPR*, 2020. 2

[50] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, and M. Kloft, "Deep semi-supervised anomaly detection," *arXiv:1906.02694*, 2019. 2

[51] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4183–4192. 2

[52] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6479–6488. 2

[53] F. V. Massoli, F. Carrara, G. Amato, and F. Falchi, "Detection of face recognition adversarial attacks," *CVIU*, 2020. 3

[54] N. Papernot and P. McDaniel, "Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning," *arXiv:1803.04765*, 2018. 3

[55] F. Carrara, F. Falchi, R. Caldelli, G. Amato, and R. Becarelli, "Adversarial image detection in deep neural networks," *Multimedia Tools and Applications*, vol. 78, no. 3, pp. 2815–2835, 2019. 3

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014. 5

[57] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv:1312.6114*, 2013. 6

[58] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves *et al.*, "Conditional image generation with PixelCNN decoders," in *NeurIPS*, 2016, pp. 4790–4798. 6

[59] D. Dehaene, O. Frigo, S. Combrexelle, and P. Eline, "Iterative energy-based projection on a normal data manifold for anomaly localization," *arXiv:2002.03734*, 2020. 7

[60] M. Sabokrou, M. Pourreza, M. Fayyaz, R. Entezari, M. Fathy, J. Gall, and E. Adeli, "AVID: Adversarial visual irregularity detection," in *ACCV*. Springer, 2018, pp. 488–505. 7

[61] F. Carrara, G. Amato, L. Brombin, F. Falchi, and C. Gennaro, "Combining GANs and autoencoders for efficient anomaly detection," *arXiv:2011.08102*, 2020. 7

[62] S. Venkataramanan, K.-C. Peng, R. V. Singh, and A. Mahalanobis, "Attention guided anomaly detection and localization in images," *arXiv:1911.08616*, 2019. 7

[63] I. Golan and R. El-Yaniv, "Deep anomaly detection using geometric transformations," in *NeurIPS*, 2018, pp. 9758–9769. 7

[64] W. Luo, W. Liu, and S. Gao, "A revisit of sparse coding based anomaly detection in stacked RNN framework," in *ICCV*, 2017, pp. 341–349. 7

[65] Y. Zhao, B. Deng, C. Shen, Y. Liu, H. Lu, and X.-S. Hua, "Spatio-temporal autoencoder for video anomaly detection," in *ACM MM*, New York, NY, USA, 2017, p. 1933–1941. 7

[66] W. Liu, W. Luo, D. Lian, and S. Gao, "Future frame prediction for anomaly detection - a new baseline," in *CVPR*, 2018, pp. 6536–6545. 7

[67] H. Park, J. Noh, and B. Ham, "Learning memory-guided normality for anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 372–14 381. 7

[68] Y. Chang, Z. Tu, W. Xie, and J. Yuan, "Clustering driven deep autoencoder for video anomaly detection," in *European Conference on Computer Vision*. Springer, 2020, pp. 329–345. 7

[69] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. 8

Dr. Fabio Valerio Massoli is a PostDoc at the Artificial Intelligence for Media and Humanities lab of ISTI-CNR. He has a Ph.D. in High Energy Physics from University of Bologna in collaboration with the Columbia University (NY). His research interests include deep learning, supervised and unsupervised learning, generative models, and quantum theory and technologies.

Dr. Fabrizio Falchi is researcher of the Artificial Intelligence for Media and Humanities lab of ISTI-CNR. He has a Ph.D. in Information Engineering from University of Pisa, and a Ph.D. in Informatics from Faculty of Informatics of Masaryk Univ. of Brno. He also received an M.B.A. from Scuola Superiore Sant'Anna in Pisa. His research interests include deep learning, convolutional neural network, similarity search, distributed indexes, multimedia information retrieval, computer vision.

Alperen Kantarci received his B.S. degree in Computer Engineering at Istanbul Technical University in 2019. He is currently pursuing M.Sc. degree in Computer Engineering at Istanbul Technical University. His research interests include computer vision, deep learning, contrastive learning and unsupervised learning.

Şeymanur Akti is a M.Sc. student and research assistant at department of computer engineering in Istanbul Technical University. She has received her B.S. degree in computer engineering from Istanbul Technical University in 2019. Her research interests include deep learning, computer vision, imbalanced data classification and anomaly detection.

Dr. Hazim Kemal Ekenel is a Professor at the Department of Computer Engineering in Istanbul Technical University. He received his PhD degree in Computer Science from the University of Karlsruhe (TH) in 2009. His research interest covers computer vision and machine learning with a focus on face analysis. He is a recipient of the Science Academy Turkey's Young Scientist Award 2018 and IEEE Turkey Section's Research Award 2019.

Dr. Giuseppe Amato was awarded a PhD in Computer Science at the University of Dortmund, Germany, in 2002. He is a senior researcher at CNR-ISTI in Pisa, where he leads the "Artificial Intelligence for Multimedia and Humanities" (AIMH) laboratory. His main research interests are artificial intelligence, content-based retrieval of multimedia documents, access methods for similarity search.