Anomaly Detection based on Zero-Shot Outlier Synthesis and Hierarchical Feature Distillation

Adín Ramírez Rivera, *Member, IEEE,* Adil Khan, *Member, IEEE,* Imad E. I. Bekkouch, Taimoor S. Sheikh

***Abstract*—Anomaly detection suffers from unbalanced data since anomalies are quite rare. Synthetically generated anomalies are a solution to such ill or not fully defined data. However, synthesis requires an expressive representation to guarantee the quality of the generated data. In this paper, we propose a two-level hierarchical latent space representation that distills inliers' feature-descriptors (through autoencoders) into more robust representations based on a variational family of distributions (through a variational autoencoder) for zero-shot anomaly generation. From the learned latent distributions, we select those that lie on the outskirts of the training data as synthetic-outlier generators. And, we synthesize from them, i.e., generate negative samples without seen them before, to train binary classifiers. We found that the use of the proposed hierarchical structure for feature distillation and fusion creates robust and general representations that allow us to synthesize pseudo outlier samples. And in turn, train robust binary classifiers for true outlier detection (without the need for actual outliers during training). We demonstrate the performance of our proposal on several benchmarks for anomaly detection.**

I. Introduction

Outlier, novelty, or anomaly detection is the process of identifying new data samples as part of the learned class (inliers) or not (outliers). This problem is relevant since most problems do not have a fully characterized set of interest data. During testing, the classifier must adapt to unseen data. There is a vast literature on these topics [1]–[4] since they have broad applications, with a particular interest in vision [5]–[11].

We can classify these methods depending on the type of modeling used, such as probabilistic-based [12]–[16], which model the distribution of inliers and the outliers are those points with low probability; distance-based [17]–[19], which identify outliers by their distance to inliers; self-representation [6], [11], [20], [21], which detect outliers as non-sparse representations on the given inliers; and deep-learning-based [9], [10], [22]–[28], which use reconstruction-based networks to detect outliers according to their reconstruction error.

Some previous methods rely on true outliers and could suffer from unbalanced data problems since anomalies are quite rare. A solution to these problems is the creation of synthetic anomalies from the anomaly class. In contrast to using true outliers for anomaly creation, we propose a zero-shot synthesis, by generating outliers from the boundaries of the inlier data, to train binary classifiers. However, using the data on the feature space is ineffective due to uncertainties on the original data and errors introduced by the feature extraction process. Instead, we propose to model the data using probability distributions through a hierarchical encoding process that models the uncertainty too. Then, we use the outskirt distributions as sources for our outlier synthesizing process. This process proves to be effective in learning a robust boundary that correctly classifies (never seen before) outliers.

Our proposed method lies at the intersection of probabilistic-, distance-, and deep-learning-based models. On one hand, we model our data (inliers) with distributions that represent the distilled information of a set of features. In contrast to existing probabilistic models, we learn a variational (autoencoder) family that is conditioned on the inlier samples alone. Thus, we generate a latent space of distributions that represent our inliers. Then, similarly to distance-based methods, we select the distributions that are on the outskirts of our learned distribution-space to generate synthetic outliers by perturbing the drawings from these distributions. Conversely to distance-based methods that work on the original feature space, our transformation creates a robust distribution-space to work with. And, instead of directly using these outskirt distributions for classification, as probabilistic or distance methods do, we synthesize outlier samples to train a binary classifier. We found that the mixture of these techniques yields better results than their traditional counterparts, with a simpler representation. We show an illustration of the overall approach in Fig. 1. Additionally, our method is trained without seeing any true outlier (thanks to our synthesis process). In a way, our synthesized outliers are distorted versions of already "strange" samples.

Our main contributions are: (1) A simple, yet powerful, zero-shot outlier synthesis framework for anomaly detection, i.e., we do not use actual outlier data to train our classifier or feature distillation but rather synthetically generated outliers from inlier data. (2) A two-level hierarchical scheme for feature distillation and creation of inlier's distributions that allows synthesizing pseudo outliers which are sufficient to train a robust binary classifier. (3) We demonstrate that the use of probabilistic modeling paired with distance-based methods for boundary definition produces robust-enough synthetic negative samples to train binary classifiers with robust performance on novel data, in particular, from an outlier class. (4) We show that loosely coupled methods have advantages over tightly coupled ones (i.e., commonly end-to-end trained methods) on a set of outlier detection benchmarks.

A. Ramírez Rivera is with the Institute of Computing, University of Campinas, Brazil, e-mail: adin@ic.unicamp.br. A. Khan, I. Bekkouch, and T. Sheikh are with the Institute of Data Science & Artificial Intelligence, Innopolis University, Russia, e-mails: {a.khan, t.sheikh}@innopolis.ru, i.bekkouch@innopolis.university.

This work was supported in part by the Brazilian National Council for Scientific and Technological Development (CNPq) under grant No. 307425/2017-7 and in part by the São Paulo Research Foundation (FAPESP) under grant No. 2019/07257-3.

Pre-print to appear in IEEE Trans. on Neural Networks and Learning Systems

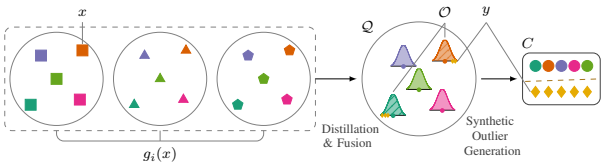


Fig. 1. Given a set of feature descriptors $\{g_i(x)\}_{i=0}^{k}$ (denoted by different shapes) for the set of inputs $x \in \mathcal{X}$ (denoted by different colors), we distill the features and fuse them into a distribution space $\mathcal{Q}$. From them, we select a set of outskirt distributions, $\mathcal{O}$, as those farther from the other set of distributions (marked with diagonal lines). They represent samples that are on the boundary of the latent space. Then, we sample, from these outskirt distributions, our synthesized-outliers $y \in \mathcal{Y}$ (denoted as diamonds). Finally, we train a classifier C using the fused inputs (inliers) and the synthesized outliers as positive and negative samples, respectively. This classifier is then used to detect true outliers, never seen before on this process.

II. Related Works

Data Representation. Existing methods rely on low level features, e.g., Histogram of Oriented Gradients [29], local patterns [30]–[34]; high level features, e.g., bag of words [35], trajectories [36]; or deep-learned features [8], [9], [23], [24], [26], [27], [37], [38]. The latter have gained traction in the past years since they tend to outperform low or high-level features designed for general purposes since the deep-features are tuned for each particular task. However, due to this same property, they are not generalizable and not generally used outside of their own proposed architecture. When a similar architecture is adapted into another problem, the previous weights need to be fine-tuned for the new tasks. On the contrary, we propose a distillation method from feature descriptors (e.g., low or high-level features, or from deep-features from other tasks) that can generalize them without adapting their original architectures. Our proposed model takes advantage of compressing the features through autoencoders (AEs). On the other hand, we also propose a zero-shot synthetic outlier generation scheme. Some methods tried to synthesize outliers while training by adding noise to the input samples [9]. Nonetheless, by perturbing the original data, they can miss the original data variations. On the contrary, we propose to synthesize data on a constructed feature-distribution space that already incorporated the variations from the inliers.

Self-Representation. By assuming that outliers cannot be reconstructed sparsely from a set of inlier points, self-representation methods [6], [11], [21] achieve outlier detection. Others exploit the correlations between the inliers that must lie on lower-dimensional spaces, and assume that incoherent samples must be outliers [20]. Another similar stream of research [14], [39] solves a problem with variations of the data points to find lower-dimensional spaces that define the inliers. The idea of sparse representation can be extended as the ability to reconstruct the samples too [6], [38]. In this sense, they are related to deep learning generative methods.

Probabilistic-based Methods. Conventional probabilistic methods [12]–[16], [40], [41] model the inliers' features' distributions, where the outliers are detected as samples with low probability. Several approximations to the true distribution of the data have been proposed, for instance, Genetic programming [42] to estimate a kernel density function, minimum-volume-sets to estimate a particular level set of the unknown nominal multivariate density [43], or constructing minimal graphs covering a K-point subset to estimate the critical region [44], [45]. Maximum-entropy-discrimination based models were proposed with several variants, and a recent method mixes this framework with hinge loss style discriminant functions and latent variables to discriminate the outliers [46]. More recent methods [24] mix the distribution's generation with manifolds to better model the inliers. Similarly, our proposed method is a mixture of ideas, and cannot be classified in a single category. On the contrary to existing probabilistic methods, our proposed method does not use the probabilities as a novelty score, but rather as a synthesizer of new data.

Distance-based Methods. The core idea of distance-based methods is the assumption that inliers are close to each other, while outliers are far from them. Commonly, k-nearest neighbors are used with density estimation [18], as well as more advanced kernel transforms [17], [19] where the distance between the projections is used for the novelty score. Conversely, we use the notion of distance to select outlier distributions that will be used later on, instead of assigning a novelty score from the neighbors. The notion of distance between points can be posed as a classification problem. One breakthrough was the kernel-based novelty detection scheme that relies on unsupervised support vector machine [47].

Deep Learning. The state of the art on outlier detection is based on deep neural networks. In particular, they continue the ideas of self-representation methods based on reconstruction. The main architecture is based on autoencoders (AEs) [22], [38]. Others [9], [10], [24], [25], [28], [48], [49] add a generative adversarial loss [50] to enhance the reconstruction of the AEs. The main idea is to train a discriminator and a generator (i.e., the AE) on a min-max fashion. This optimization scheme yields better results for the generation of the original samples. However, it is hard to train, and it is unstable due to the different learning rates of its components. Similarly to existing approaches, we use the unsupervised nature of AEs to learn a compact space of the original features (akin to self-representation). However, we do not work on this space, and, yet again, we produce a higher-level feature space based on distributions through variational autoencoders (VAEs), which have shown excellent capabilities for unsupervised clustering [51]–[55]. And in contrast to common deep learning approaches, we detect outliers with vanilla classifiers.

Novelty. We use AEs and VAEs similar to previous approaches [53], [56]. (1) However, our proposal's novelty lies in the construction of a two-level latent space hierarchy that (i) encodes the inliers into a low-dimensional latent space that supports a reliable estimate on inliers' distributions (through AEs); and (ii) provides a dense representation of the points in a distribution-space where inlier distributions can be compacted to find a reliable boundary (through VAEs). (2) Unlike previous tightly coupled outlier detectors [57], we separate the feature modeling, synthetic-outlier generation, and classification, which enables our learning spaces to be used on a diverse set of tasks. (3) Our zero-shot outlier generation from near-boundary inlier-distributions is different from previous generation methods [58] that commonly assume a uniform maximum entropy distributed

outlier class. Our method is superior in the sense that we do not assume any distribution over the outliers. Instead, we rely only on the inliers' distributions (available at training time), and synthesize outliers from them (at training time). Conversely to probabilistic methods that rely on matching different inlier and outlier distributions, we use a distance-based mechanism (training a classifier on synthetic outliers sampled from the distributions and the true inliers) to classify outliers and inliers (since the distributions and the outliers themselves are unknown at training time).

III. Proposed Method

We define outlier detection as a binary classification problem. Such that given any sample from the inlier set $x \in \mathcal{X}$, we classify it as inlier, while any other data $y \notin \mathcal{X}$ is classified as an outlier. However, we assume that the outliers are unknown at training time, which allows us to tackle broader problems. This setup poses a challenge since we need negative samples to train the binary classifier. Although a one-class classifier may be an option, we show that learning a good boundary outperforms it (cf. Section IV).

To train a binary classifier using only the inlier data, we need to synthesize outliers to act as negative samples. We find the boundary of the training data, and then synthesize the outlier samples from it (detailed on Section III-B). Instead of working on the original data or feature spaces, we learn a distribution-space that represents the data changes and uncertainties (see Section III-A). This feature distillation process helps to find better boundaries between inliers and outliers. We detail our proposed hierarchical encoding scheme in Section III-C.

A. Probabilistic Representation

We are interested in modeling the inliers as distributions to consider their uncertainty, in addition to compressing their representation. Thus, we create a conditioned Gaussian distribution that represents each data point by learning a variational autoencoder (14), see Section III-C for details. Given an inlier $x \in \mathcal{X}$ and its latent vector $w(x)$ (13), we learn a distribution conditioned on the latent representations of the given data point $q\left(z \mid w(x)\right) \equiv (\mu, \sigma)$. In practice, we use standard Gaussians to represent the distributions, and we are interested in obtaining one distribution per data point in our training set. In other words, we construct the set of distributions

$$\mathcal{Q} = \{(\mu, \sigma) = q\left(z \mid w(x)\right) : x \in \mathcal{X}\}. \tag{1}$$

That is, each distribution $q \in \mathcal{Q}$ is a Normal distribution with parameters equal to the latent representation of our VAE that corresponds to a given inlier x in the training set.

B. Outlier Classification

Our synthesis process consists of perturbing inliers that are in the boundary of the training data, and using them with the inliers to train our classifier. We perturb the inliers by drawing from the distributions that correspond to the borderline inliers. In other words, our goal is to find the distributions from $\mathcal{Q}$ (1) that are on the outskirts of the space (as shown in Fig. 1). Our first task is to find the distribution-space's boundary given the training inliers. We propose two methods to define this boundary and its corresponding distributions.

Ellipsoid-based. The first boundary-defining method is based on selecting the distributions that lie on the boundary of the distribution of distributions (i.e., a meta-distribution). To find them, we compute the center, $\bar{\mu}$, and the standard deviation, $\bar{\sigma}$, of the meta-distribution of $\mathcal{Q}$ through

$$\bar{\mu} = \frac{1}{|\mathcal{Q}|} \sum_{i=1}^{|\mathcal{Q}|} \mu_i, \tag{2}$$

$$\bar{\sigma}^2 = \frac{1}{|\mathcal{Q}| - 1} \sum_{i=1}^{|\mathcal{Q}|} (\mu_i - \bar{\mu})^2, \tag{3}$$

where $|\mathcal{Q}|$ is the size of the set that represents the amount of samples. Then, our outskirt distributions are within the set

$$\mathcal{O}_e = \left\{ (\mu, \sigma) : \sum_i \frac{(\mu_i - \bar{\mu}_i)^2}{\alpha \bar{\sigma}_i^2} \geq 1, (\mu, \sigma) \in \mathcal{Q} \right\}, \tag{4}$$

where the subscripted values represent the components of the vectors, and α is a scaling hyperparameter. In other words, we select the outskirt distributions as those that lie outside of the ellipsoid defined by the scaled (by α) mean and variance of the meta-distribution. Note that we constructed a diagonal covariance matrix through the above procedure.

Distance-based. On the other hand, since we have high dimensional distributions, we would be computing their distributions on a very sparse space. Instead, we propose to analyze the ℓ_2 norm of their means. Consequently, we collapse the high dimensional space into a lower one, and then we proceed in a similar fashion as above. We compute the mean, $\bar{\mu}_\ell$, and the standard deviation, $\bar{\sigma}_\ell$, of the ℓ_2 norms ($\|\cdot\|_2$) of all the distributions' means on $\mathcal{Q}$, such that

$$\bar{\mu}_\ell = \frac{1}{|\mathcal{Q}|} \sum_{i=1}^{|\mathcal{Q}|} \|\mu_i\|_2, \tag{5}$$

$$\bar{\sigma}_\ell^2 = \frac{1}{|\mathcal{Q}| - 1} \sum_{i=1}^{|\mathcal{Q}|} (\|\mu_i\|_2 - \bar{\mu}_\ell)^2. \tag{6}$$

And the outskirts distributions are within the set

$$\mathcal{O}_\ell = \left\{ (\mu, \sigma) : \left| \|\mu\|_2 - \bar{\mu}_\ell \right| \geq \alpha \bar{\sigma}_\ell, (\mu, \sigma) \in \mathcal{Q} \right\}. \tag{7}$$

Similarly, the α parameter scales the boundary of which distributions will be selected.

Outlier Synthesis. Finally, given any set of distributions that lie on the outskirts of the space, $\mathcal{O}$, we can synthesize a set of outlier points that will be our negative class for training. Hence, let the set of synthetic outlier samples be

$$\mathcal{Y} = \{ y = \mu + \beta \sigma \epsilon : (\mu, \sigma) \in \mathcal{O}, \epsilon \sim \mathcal{E} \}, \tag{8}$$

where each new sample is drawn from the distributions defined in $\mathcal{O}$, ϵ is generated noise from a distribution $\mathcal{E}$, and we use a scalar factor β to draw from the tails of the distribution. Geometrically, we can draw from any direction from the mean. However, we are interested in the samples that lie on the outside part of the boundary. E.g., in Fig. 1, we draw samples on the

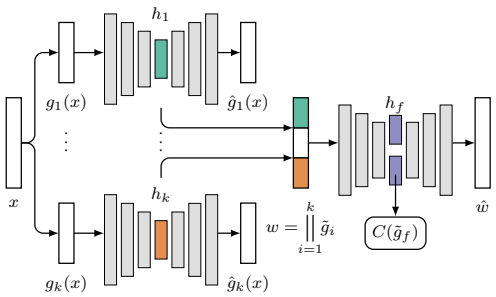


Fig. 2. Our feature distillation architecture uses k features $\{g_i(x)\}_{i=0}^{k}$ from the original data point x, and process them through an AE, h_i, for it to learn an approximated version of them, $\hat{g}_i(x)$. Simultaneously, we concatenate the encoded features of each AE, h_i, namely $\tilde{g}_i(x)$, into a vector w. Then, we pass it through a learned fusion function, h_f, that is a VAE to recover the input, $\hat{w}$. The VAE encodes the parameters of the distribution of w, $(\mu, \sigma) \equiv q(\cdot \mid w)$. The set of these parameters for all data points is $\mathcal{Q}$. We use samples from the encoder of h_f, namely $\tilde{g}_f$, as the positive inputs for a classifier C, while the negative samples come from the synthesized outliers from the distributions at the outskirts (cf. Section III-B).

side closer to the boundary of each of the outskirt distributions instead of sampling from any side. Hence, we construct β such that for each dimension its value is positive or negative depending on whether the mean of this distribution is greater or less than the mean of the meta-distribution. We uniformly sample from all of the distributions to create enough outliers to match the amount of inliers used for training.

Classification. Once we have defined our training sets, $\mathcal{X}$ and $\mathcal{Y}$, we can proceed to train a classifier C by considering the projected $\mathcal{X}$ in our distribution space (in practice, we use the mean of the corresponding distribution since they represent the expected value of each inlier) as our positive examples, and $\mathcal{Y}$ as our negative ones. During testing, we introduce new (never seen) samples to the classifier to evaluate its capabilities for outlier detection. We present more details on Section IV regarding our experimental setup. Note that we defined a loosely coupled process that allows us to switch between classifiers, without the need to relearn the feature distillation process.

C. Feature Encoding

Our model for anomaly detection relies on a hierarchical feature extraction and fusion scheme based on AEs (as shown in Fig. 1). Our proposal uses a two-level contraction and fusion method, as shown in Fig. 2. Lets consider the raw data $x \in \mathcal{X}$, and a extracted set of k features $G = \{g_i(x)\}_{i=1}^{k}$. Our goal is to create a mixture of the first level features G that may not have all the characteristics of x. For instance, we may be using feature descriptors that are easy to compute but not expressive enough, or features learned from different tasks that need to be fused to have a more powerful descriptor, among other cases. Hence, if we convert this data into a more suitable representation that has particular distributions or characteristics, it will be easier to classify. Several methods can be used to achieve such space, among them, one can do dimensionality reduction or clustering followed by a mapping function. Nevertheless, AEs have demonstrated these capabilities within a single method. They learn an identity function with the property of embedding the input vectors into a lower-dimensional space that distills the relevant features. Due to this property, they are the backbone of our proposed architecture.

The first level of our model consists of a set of k AEs, $H = \{h_i^{\theta_i}\}_{i=1}^{k}$, parameterized by $\theta_i \in \Theta$, where Θ is the set of parameters for the whole model. (In the following, we suppress the parameters from the notation of each AE for brevity.) The job of each AE is to compact the input g_i into a more expressive space, that is,

$$\hat{g}_i(x) = h_i\left(g_i\left(x\right)\right), \tag{9}$$

the ith reconstructed feature $\hat{g}_i(x)$ is the output of the ith AE h_i that operates on the input feature $g_i(x)$. During training, one of our objectives is to minimize the error between the reconstructed and the original features, that is our feature loss

$$\mathcal{L}_f = \sum_{i=1}^{k} \left\| \hat{g}_i(x) - g_i(x) \right\|^2. \tag{10}$$

Moreover, we are interested in the compact representation learned within the AEs, as they define a space that clusters and separates the input data. The ith AE, h_i, is the composition of encoder, E_i, and decoder, D_i, functions such that

$$h_i(z) = D_i\left(E_i\left(z\right)\right). \tag{11}$$

Lets denote the inner compact representation of the ith AE (the output of the encoder) as

$$\tilde{g}_i(x) = E_i\left(g_i(x)\right). \tag{12}$$

Then, we are interested in the learned representation

$$w(x) = \Big\|_{i=1}^{k} \tilde{g}_i(x), \tag{13}$$

where $\|$ is the concatenation operator. Our objective is to learn a transformation function, h_f, that converts this distilled representation into a single compact space. Since having a sparse vectorial representation, through the AEs, is cumbersome, we plan to learn a continuous space by modeling each vector as a distribution. Hence, we use variational inference to learn a variational family approximated through a VAE, defined by

$$\hat{w} = h_f(w), \tag{14}$$

where h_f comprises the encoding of w into a distribution $q(z \mid w) = E_f(w)$, and the decoding through a sampling process $\hat{w} \sim q(z \mid w)$. (We omit the argument x from w for brevity.) And, we define the loss for the high level features as

$$\mathcal{L}_h = \|\hat{w} - w\|^2 + \mathrm{KL}(q(z \mid w) \parallel \mathcal{N}(0, I)), \tag{15}$$

where KL is the Kullback-Leibler divergence to measure the distance between the prior and the learned distribution. We assume that our proposed distribution, $q(z \mid w)$, is distributed as a Gaussian and try to minimize its distance towards a zero-centered unitary Gaussian, $\mathcal{N}(0, I)$.

Finally, our objective is to find the set of parameters, Θ, of the neural networks (the set $\{H, h_f\}$) that minimizes the sum of the loss functions

$$\Theta^* = \arg\min_{\Theta} \left\{ \mathcal{L}_f^{\Theta} + \mathcal{L}_h^{\Theta} \right\}, \tag{16}$$

where the loss functions, (10) and (15), are parameterized by Θ which corresponds to the parameters of AEs and VAE, $\{H^{\Theta}, h_f^{\Theta}\}$. These parameters are learned through back-propagation on the training phase.

IV. Results and Evaluation

For our experiments we are evaluating our method for outlier generation based on two different ways of selecting the outlier-generation distributions, i.e., (4) and (7), that we named OG_e and OG_ℓ, respectively. We selected three features, namely Histogram of Oriented Gradients (HoG) [29], Local Binary Patterns (LBP) [59], and raw pixel information as our set of features descriptors for our hierarchical learning. For the classifier C, we are using a Support Vector Machine, due to training easiness. Nevertheless, any amount of features and other classifiers can be used. We used the area under the curve (AUC) of the Receiver Operating Characteristic curve, and F_1-score, which is the weighted average of the precision and recall, as our evaluation metrics for all experiments. In both cases, the higher the score the better the methods. The results reported in this section are from our implementation using Python 3.6.5 executed on an NVIDIA GeForce GTX 1080 Ti with the TensorFlow framework. We detail the neural network architecture and the feature extractors hyperparameters on the supplementary material.

A. Experimental Setup

We evaluated the performance of our proposed method using MNIST [60], [61], Caltech [62], Coil-100 [63], Extended Yale B [64], [65], UCSD Pedestrian [66], ISIC 2018 Challenge (Task 3) [67], [68], and CIFAR-10 [69] datasets.

We followed the same experimental setup as previous works [9], [11] for MNIST, Caltech, Coil-100 and Extended Yale B. For UCSD Pedestrian and ISIC 2018 Challenge (Task 3) datasets, we followed Chaker et al.'s [70] and Lu and Xu's [71] setups, respectively. For CIFAR-10, we followed previous methods [72]–[74] setups.

In general, we selected inliers from a set of classes of the dataset, and the evaluation outliers were randomly sampled from the remaining classes. We describe the setup of each experiment per dataset on the supplementary material. For each of our methods and per experiment, we perform a train, validation, and test (inlier) data split. To find the best hyperparameters of the model, we used the train and validation split. And, we evaluated in the test partition. We report the average metrics on the test partition of 5-fold cross-validation for the inlier data split. Note that our training data contains no true outliers. We train the classifier using our synthesized outliers as negative samples, as previously described, and then evaluated, during testing, on different percentages of true outliers (according to the setups of each experiment).

B. Comparison against Baselines

For our experiments on MNIST, Caltech-256, Coil-100, and Yale B, we compared our proposed methods against a set of methods, namely LOF [18], DRAE [10], R and RD [9], GPND [24], Coherent Pursuit (CoP) [20], REAPER [14], Outlier Pursuit (OP) [39], LRR [75], DPCP [76], ℓ_1-thresh. [77], R-Graph [11], AnoGAN [27], and AGAN [23].

TABLE I. Comparison of metrics on the MNIST dataset when varying the percentage (%) of the evaluation data comprised by outliers.

	%	LOF	DRAE	D	RD	GPND	OG_e	OG_ℓ
AUC	10	–	–	–	–	–	0.9918	**0.9920**
F_1		0.92	0.95	0.93	0.97	0.96	0.9919	**0.9921**
AUC	20	–	–	–	–	–	0.9912	**0.9915**
F_1		0.83	0.91	0.90	0.92	0.95	0.9913	**0.9916**
AUC	30	–	–	–	–	–	0.9912	**0.9914**
F_1		0.72	0.88	0.87	0.92	0.94	0.9913	**0.9915**
AUC	40	–	–	–	–	–	0.9911	**0.9913**
F_1		0.65	0.82	0.84	0.91	0.93	0.9912	**0.9914**
AUC	50	–	–	–	–	–	0.9911	**0.9914**
F_1		0.55	0.73	0.82	0.88	0.93	0.9912	**0.9915**

In the MNIST and Caltech-256 experiments, our proposed methods obtained the highest results while maintaining a stable outlier detection rate regardless of the number of outliers, as shown in Tables I and II. Moreover, our proposed method learns representations that are robust to the type of classes, as demonstrated by the results when showing them more classes in the Caltech-256 experiment (cf. Table II). In contrast, for the Coil-100 and Yale B experiments, which present variations, such as rotation and illumination, respectively, our proposed methods do not outperform R-Graph according to the AUC, as shown in Tables III and IV. Interestingly, we consistently obtain higher F_1 scores. Moreover, we also noted that the variants introduced in these databases pose a great challenge for the descriptors we are using since HoG and LBP are not invariant to the point of view and intense intensity transforms of these two datasets. Nevertheless, our method still achieves a comparable performance with existing methods.

In Table V, we reported the AUC performance of our proposed method on UCSD Pedestrian benchmark with respect to other methods, namely, Adam [78], SF [79], MPPCA [80], DTM [81], MPPCA-SF [81], Sparce [6], SNM [70], LSR-VAE [82], and GMFC-VAE [83]. OG_e outperforms the other methods in Ped2 and detects different kinds of anomalous events, such as bicycling, skateboarding, wheel-chair, etc. Whereas in Ped1 our methods achieve comparable results to LSR-VAE [82] but 3% to 4% lower than those reported by Fan et al. [83]. We noticed that in Ped1 are several illumination variants when groups of people walk towards and away from the camera, and some amount of perspective distortion which affects the selected features.

The AUC of our experiments on the ISIC dataset are summarized in Table VI. The highest AUC and more stable results were obtained by ℓ_2-based selection method as compared to ellipsoid with an overall AUC score of 87.2%. In general, our proposal outperforms other methods [71], except on the MEL disease. We assume that the problems are due to variations on the images that are not picked up, like in previous errors.

C. Ablation Study

1) Outlier Synthesis' Parameters: For the parameters that define the selection of distributions at the outskirts of the

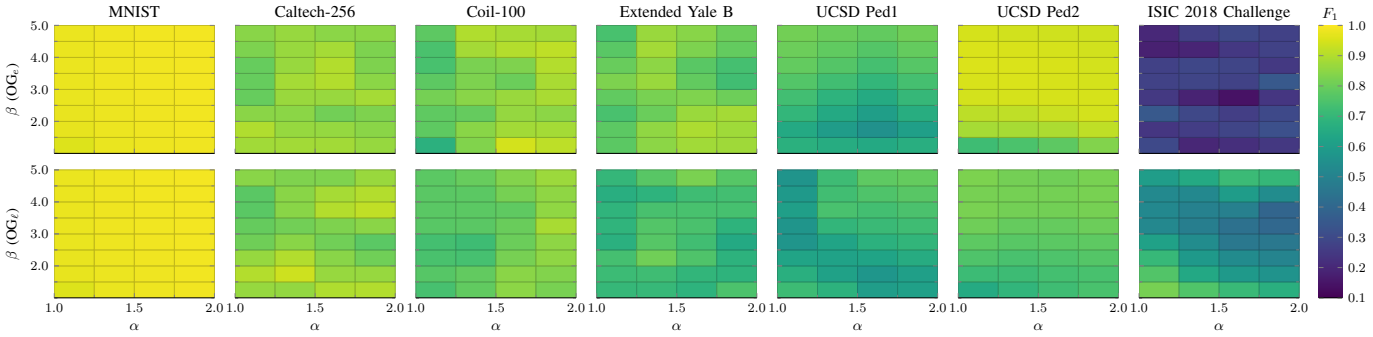


Fig. 3. Comparisons of F_1 scores on MNIST, Caltech-256, Coil-100, Extended Yale B, UCSD Ped1 and Ped2, and ISIC 2018 Challenge datasets by varying α and β parameters when inliers come from one class and true outliers from remaining classes are used for testing (selected randomly from the remaining classes).

TABLE II. Results on the Caltech-256 dataset. Each pair of rows shows the results when inliers were taken from 1, 3, and 5 categories. For evaluation, 50% of the data were outliers sampled from the category 'clutter.'

	CoP	REAPER	OP	LRR	DPCP	ℓ_1-thres.	R-Graph	D	RD	AGAN	AnoGAN	OG$_e$	OG$_\ell$
AUC	0.905	0.816	0.837	0.907	0.783	0.772	0.948	0.932	0.942	–	–	**0.991**	0.989
F_1	0.880	0.808	0.823	0.893	0.785	0.772	0.914	0.916	0.928	0.977	0.956	**0.991**	0.988
AUC	0.676	0.796	0.788	0.479	0.798	0.810	0.929	0.930	0.938	–	–	0.996	**0.997**
F_1	0.718	0.784	0.779	0.671	0.777	0.782	0.880	0.902	0.913	0.963	0.915	0.996	**0.997**
AUC	0.487	0.657	0.629	0.337	0.676	0.774	0.913	0.913	0.923	–	–	**0.998**	0.997
F_1	0.672	0.716	0.711	0.667	0.715	0.762	0.858	0.890	0.905	0.945	0.887	**0.998**	0.997

TABLE III. Results on the Coil-100 database. Each pair of rows shows the results when inliers were taken from 1, 4, and 7 categories with 50%, 25%, and 15% of outlier samples on the evaluation set. Inliers categories were randomly chosen, and outliers were randomly sampled, in the aforementioned proportions, from the remaining categories (at most one from each category).

	CoP	REAPER	OP	LRR	DPCP	ℓ_1-thres.	R-Graph	GPND	OG$_e$	OG$_\ell$
AUC	0.843	0.900	0.908	0.847	0.900	0.991	**0.997**	0.968	0.972	0.979
F_1	0.866	0.892	0.902	0.872	0.882	0.978	**0.990**	0.979	0.972	0.979
AUC	0.628	0.877	0.837	0.687	0.859	0.992	**0.996**	0.945	0.982	0.988
F_1	0.500	0.703	0.686	0.541	0.684	0.941	0.970	0.960	0.982	**0.987**
AUC	0.580	0.824	0.822	0.628	0.804	0.991	**0.996**	0.919	0.979	0.975
F_1	0.346	0.541	0.528	0.366	0.511	0.897	0.955	0.941	**0.978**	0.975

space, α, (4) and (7), and β (8), we evaluated the ranges of α from 1 to 2 with step 0.25, and β from 1 to 5 with step 0.5, on the best set of hyperparameters found on previous experiments, cf. Section IV-B. We show in Fig. 3 the results for our metrics on all our evaluation datasets when using true outliers from the remaining classes on testing. We show a detailed evaluation with other setups for outliers on all the datasets on the supplementary material.

We observed that selecting outskirt distributions that are more than one standard deviation away from the center of the meta-distribution yields better results, i.e., having higher values for α. This trend can be seen in all datasets. However, it is clearer in Caltech, Coil and UCSD Pedestrian datasets, and a dip can be seen on MNIST when both parameters are one. On the other hand, sampling the synthetic outliers farther from the center of the distributions, as indicated by higher β values, shows an improvement. However, different from α, the variations in β are not consistent as the former, as β's metrics have falls on the different α values. We hypothesize that, in general, these parameters will present an inverted-u behavior since they control how different the synthesized samples could be. By increasing them, we will increase the performance up to a certain point, and then we will get diminishing returns. Note

TABLE IV. Results on the Extended Yale B database. Each pair of rows shows the results when inliers were taken from 1 and 3 randomly chosen subjects, and outliers were randomly chosen from the other subjects (at most one from each subject).

	CoP	REAPER	OP	LRR	DPCP	ℓ_1-thres.	R-Graph	OG$_e$	OG$_\ell$
AUC	0.556	0.964	0.972	0.857	0.952	0.844	**0.986**	0.976	0.957
F_1	0.563	0.911	0.918	0.797	0.885	0.763	0.951	**0.975**	0.954
AUC	0.529	0.932	0.968	0.807	0.888	0.848	**0.985**	0.939	0.943
F_1	0.292	0.758	0.856	0.509	0.653	0.545	0.878	0.933	**0.943**

TABLE V. Results on the UCSD Pedestrian 1 and 2 datasets with the comparison of AUC performance where inliers were pedestrians and outliers were other objects.

	Adam	SF	MPPCA	DTM	MPPCA-SF	Sparce	SNM	LSR-VAE	GMFC-VAE	OG$_e$	OG$_\ell$
Ped1	–	0.675	0.590	0.818	0.668	0.860	0.855	0.902	**0.949**	0.912	0.908
Ped2	0.634	0.623	0.774	0.848	0.710	0.861	0.879	0.891	0.922	**0.955**	0.900

that high values of α introduce outliers within the training inliers, while high values on β may introduce outliers that are too different from any expected anomaly at prediction time. Hence, a classifier trained on this data will learn a decision boundary that is not close enough to the normal samples.

We note the particular behavior of the metrics on Coil-100, Extended Yale-B, and ISIC datasets. On these datasets, there are several dips and changes on the F_1 scores. Moreover, no patterns seem to arise from the experiments. We notice, however, that the selection of outskirts distributions varies considerably. We attribute this behavior to two factors: the variations on the datasets, and the amount of data. On one hand, the feature descriptors we used are sensitive to the variations of these two datasets, which may be reflected in the changes in the selection of the outliers (cf. ablation results on Section IV-C2). On the other hand, these datasets have considerably fewer samples, in comparison with MNIST, Caltech-256, and UCSD Pedestrian. For example, MNIST varies between 5500–5900 samples, Caltech has 750 (when $n = 5$), in comparison Coil-100 has 504 samples (when $n = 7$) and Yale B has only 192 samples (when $n = 3$). Note the variation in the classes and their relation to the number of samples of each class.

2) Evaluation of Other Dimensions: We may think of our proposed method in terms of four different dimensions: features used, way of applying the AEs on them, the fusion through the VAE, and the outskirt-distribution selection method. To assess our method we performed a set of experiments by varying these dimensions on the Extended Yale B dataset, and we

TABLE VI. Results on the ISIC 2018 Challenge (Task 3) dataset. Each row shows the AUC when inliers were taken from NV disease, and row's category was used as an outlier.

Disease	$s_{\text{vae}}^{\text{reconst}}$	$s_{\text{iwae}}^{\text{reconst}}$	$s_{\text{vae}}^{\text{kl}}$	$s_{\text{iwae}}^{\text{kl}}$	s_{vae}	s_{iwae}	OG_e	OG_ℓ
AKIEC	0.872	0.871	0.441	0.406	0.864	0.864	0.727	**0.943**
BCC	0.803	0.802	0.454	0.431	0.795	0.795	0.708	**0.872**
BKL	0.792	0.793	0.472	0.441	0.783	0.784	0.772	**0.934**
DF	0.682	0.678	0.398	0.383	0.671	0.670	0.741	**0.916**
MEL	0.862	**0.864**	0.690	0.677	0.861	0.861	0.736	0.845
VASC	0.662	0.657	0.487	0.477	0.651	0.648	0.716	**0.840**
All Disease	0.779	0.777	0.491	0.469	0.771	0.771	0.725	**0.872**

report the results in Table VII. We selected this dataset for two reasons: size, which allows us to run several experiments on our restricted hardware, and challenging variations, which gives us a lower bound on the expected results.

For the features, we evaluate the combination of HoG (H), LBP (L), and raw intensity (R) as pairs and triplets. For the AEs, we evaluate the way of applying them to the set of inputs. We can use different AEs per feature, i.e., individually to each feature (I), use a single AE on the concatenation of the inputs (C), or use no AE at all (WO). Moreover, we consider the setup with (W) and without (WO) VAE for fusion. In the latter setup, the classification is performed over the concatenation of the extracted representations, instead of the fusion. Lastly, we consider the use of our two proposed outskirt selection schemes, i.e., ellipsoid (e) and ℓ_2-based (ℓ). In the case where no distributions are generated (i.e., the case without VAE for fusion), we cannot use the selection method and the concatenated features are passed to the classifier. The cases with no synthesis are denoted with an n-dash (–). In them, we use a one-class SVM as a classifier due to the lack of negative samples to train a binary one.

In general, we observe that fusion is a critical component of the proposed method. Removing the VAE (WO in the VAE column) has a significant impact on the different results regardless of the combination of the AE and features. We, also, compared the use of an AE instead of VAE with and without synthetic outlier generation by randomly jitter the outskirt features in the AE space with noise drawn from $\mathcal{N}(0, 1.5)$, see the end of Table VII. These results show a significant decrease in performance. On the other hand, the second dimension that affects the results is the way we apply the AEs. By encoding each feature individually we obtain consistently higher metrics. These results align with the objective of encoding and compressing each feature individually to later fuse them. By having an early fusion (the concatenation at the AE level), the VAE does not have richer spaces to work with.

To evaluate the contribution of the generation of the synthetic outliers, we trained a one-class SVM using only inlier data with different feature combinations (denoted by the n-dash on the OS column). We notice a significant drop in the metrics concerning their counterparts that use synthetic outliers for training. Moreover, we also evaluate the case in which the distribution-space is used as features for classification (denoted by '[H][L][R] C W –' rows). Similarly, we can see that the performance comes from the synthetic outliers that help the classifier to learn a robust boundary, in contrast to the use of more expressive features.

TABLE VII. Ablation study on the Extended Yale B database, where inliers are taken from one and three (n) subjects with 50% and 15% outlier samples from the other subjects (at most one from each subject) all randomly chosen. We used different combinations of features, i.e., HoG (H), LBP (L), and raw intensity (R); way of applying the AE, such as individually to each feature (I), or to the concatenated features (C), or without AE and passing the features to the next step (WO); existence of VAE, such as with (W) and without (WO) it; and different outskirt distribution selection (OS), i.e., ellipsoid (e), ℓ_2-based (ℓ), and without it (–). The hat OS denote experiments when the outliers were generated deterministically. The last section show results with AE instead of VAE with outliers generated with jitter and without them (–). Blanks in a cell denote that the previous row's setup was used.

Feats.	AE	VAE	OS	$n=1$		$n=3$	
				AUC	F_1	AUC	F_1
HLR	I	W	e	0.976	0.975	0.939	0.933
			ℓ	0.957	0.954	0.943	0.943
HL			e	0.883	0.884	0.897	0.902
			ℓ	0.914	0.905	0.903	0.904
HR			e	0.950	0.946	0.938	0.934
			ℓ	0.957	0.955	0.908	0.897
LR			e	0.957	0.955	0.884	0.884
			ℓ	0.930	0.923	0.898	0.891
HLR	C	W	e	0.912	0.901	0.912	0.904
			ℓ	0.908	0.897	0.927	0.923
HL			e	0.922	0.915	0.902	0.919
			ℓ	0.828	0.833	0.886	0.878
HR			e	0.908	0.898	0.904	0.893
			ℓ	0.943	0.940	0.886	0.871
LR			e	0.943	0.939	0.912	0.903
			ℓ	0.927	0.920	0.911	0.901
HLR	WO	W	e	0.925	0.918	0.923	0.930
			ℓ	0.905	0.892	0.911	0.919
HL			e	0.810	0.817	0.802	0.795
			ℓ	0.877	0.890	0.833	0.825
HR			e	0.891	0.891	0.907	0.910
			ℓ	0.939	0.934	0.908	0.904
LR			e	0.931	0.925	0.901	0.899
			ℓ	0.890	0.893	0.927	0.921
HLR	C	W	–	0.621	0.610	0.611	0.621
HL				0.662	0.764	0.614	0.450
HR				0.632	0.725	0.576	0.407
LR				0.593	0.655	0.610	0.447
HLR	I	WO	–	0.489	0.616	0.619	0.277
HL				0.550	0.560	0.481	0.171
HR				0.633	0.676	0.578	0.247
LR				0.551	0.567	0.527	0.210
HLR	C	WO	–	0.500	0.505	0.568	0.245
HL				0.580	0.601	0.544	0.219
HR				0.509	0.489	0.590	0.249
LR				0.586	0.623	0.580	0.254
HLR	WO	WO	–	0.413	0.394	0.492	0.183
HL				0.413	0.394	0.450	0.177
HR				0.514	0.563	0.414	0.179
LR				0.413	0.394	0.507	0.195
HLR	I	W	$\hat{e}$	0.601	0.621	0.611	0.575
			$\hat{\ell}$	0.600	0.622	0.611	0.575
HLR	I	AE	–	0.542	0.643	0.676	0.391
HL				0.604	0.631	0.517	0.284
HR				0.641	0.683	0.609	0.314
LR				0.589	0.614	0.592	0.284
HLR	C	AE	–	0.523	0.535	0.591	0.308
HL				0.584	0.656	0.572	0.294
HR				0.557	0.540	0.635	0.314
LR				0.619	0.647	0.603	0.346
HLR	I	AE	jitter	0.849	0.834	0.825	0.811
HL				0.768	0.759	0.796	0.814
HR				0.831	0.793	0.826	0.807
LR				0.812	0.828	0.771	0.793
HLR	C	AE	jitter	0.727	0.749	0.741	0.784
HL				0.704	0.697	0.752	0.710
HR				0.674	0.691	0.735	0.724
LR				0.765	0.751	0.718	0.682

Interestingly, the results of HR or LR features show higher metrics in comparison to HL features. These results indicate, again, that fusion can do so much to increase the invariants of the given inputs' features. We can draw similar conclusions from the results of our proposed method and more complex

TABLE VIII. Comparison of the proposed method against VAE and AE when trained on true outliers (upper bound performance). We report the AUC scores of all the methods on 5-fold cross-validation with 20% outliers. Our proposal was trained with synthetic data, but cannot be evaluated as a one-class due to lack of outliers. VAE and AE were trained on both binary (first four rows) and one-class (last two rows) setups. We report max-min values in parenthesis.

	Dataset		
	CIFAR 10	Extended Yale B	Coil-100
VAE	0.937 ± 0.012 (0.957, 0.919)	0.987 ± 0.019 (0.998, 0.965)	0.997 ± 0.022 (0.999, 0.952)
AE	0.929 ± 0.009 (0.941, 0.914)	0.965 ± 0.014 (0.984, 0.937)	0.989 ± 0.007 (0.997, 0.974)
OG_e	0.883 ± 0.031 (0.925, 0.832)	0.971 ± 0.026 (0.998, 0.920)	0.985 ± 0.032 (0.998, 0.916)
OG_ℓ	0.915 ± 0.023 (0.963, 0.878)	0.983 ± 0.016 (0.998, 0.953)	0.991 ± 0.012 (0.998, 0.959)
VAE (OC)	0.504 ± 0.002 (0.538, 0.480)	0.671 ± 0.022 (0.706, 0.634)	0.728 ± 0.019 (0.740, 0.687)
AE (OC)	0.502 ± 0.002 (0.535, 0.462)	0.636 ± 0.016 (0.661, 0.591)	0.711 ± 0.013 (0.749, 0.684)

fusion (cf. first two rows of Table VII and Section IV-B). However, our proposed method can boost the metrics significantly on this task, in comparison to other methods (including concatenation and learning on the original space).

We also evaluated the case where the outliers are generated deterministically instead of using our probabilistic sampling. In this case, we set $\epsilon = 1$ in the generation (8). In that way, we will always generate the same outlier per outskirt distribution. The corresponding deterministic counterparts of our outskirt selectors, (4) and (7), are denoted as $\hat{e}$ and $\hat{\ell}$, respectively. The results, in the last rows of the table, show a significant reduction in the different metrics. This demonstrates the importance of creating robust outliers for training when lacking negative samples, independently of the robust feature spaces.

In summary, the way of fusing the feature spaces is a key component in the final representation. And our probabilistic approach seems to yield excellent results in comparison to naive fusion methods, and existing approaches that do not fuse. Secondly, we need to represent spaces individually to take advantage of each feature space's characteristics. Moreover, a probabilistic representation has the advantage of synthesizing robust outliers, which are necessary to learn robust and general classifiers. Regarding the outlier selection proposals, we could not draw a line of which one is best for the task since no pattern emerged from the results.

3) Comparison against Gold Standard: We also compared our proposal against what can be considered an upper bound of performance of the method. That is, we compared against a VAE and an AE with the same architecture as the proposal, except that they were trained on true outliers. We show the results on Table VIII. We can observe that our proposal is close to the upper bound that represent the VAE and the AE results. It is expected that the proposal achieves a lower performance due to the lack of true outliers. Moreover, we note that the AE achieves a lower performance than its variational counterpart which also supports our decision to use a VAE as the main feature descriptor.

D. Evaluation of Other Classifiers

Our previous experiments used SVM as the classifier, in this section, we explore the impact of other classifiers on our method's performance. Namely, we additionally use Multilayer Perceptron (MLP), Random Forest (RF), and Naive Bayes (NB). Simultaneously, we evaluate our method against a set of algorithms that report results under a different dataset and metric. We compare against One Class SVM (OC-SVM), Deep Autoencoding Gaussian Mixture Model (DAGMM), Deep structured energy-based models (DSEBM), and Anomaly Detection Transformations (ADT) reported by Golan and Ran [72]; three ADT-based models, and the COREL method reported by Hofer et al. [73]; and AnoGAN and Deep-SVDD reported by Ruff et al. [74]. In this case, we evaluate our results using the area under the curve of the ROC (AUC) on the CIFAR-10 dataset. This change in metric and dataset is to compare against these other sets of methods. There are more challenging datasets and versions of the CIFAR dataset. However, not all methods report on them. Hence, it is difficult to have a comparison against more recent methods based on those other datasets.

TABLE IX. AUC scores of ellipsoid (e), ℓ_2-based (ℓ) methods on CIFAR-10 dataset where each model was trained on a single class and tested against all other classes. We report averaged results overall classes with 5-fold cross-validation, and different classifiers: Support Vector Machine (SVM), Multilayer Perceptron (MLP), Random Forest (RF), and Naive Bayes (NB). We compared our AUC against other existing methods.

Feats.	OS	SVM	MLP	RF	NB
HLR	ℓ	0.917 ± 0.026	0.813 ± 0.026	0.958 ± 0.029	0.886 ± 0.013
	e	0.858 ± 0.022	0.772 ± 0.045	0.836 ± 0.014	0.856 ± 0.018

OC-SVM (CAE) [72]	0.624	OC-SVM (E2E) [72]	0.648
OC-SVM (RAW) [72]	0.620	DAGMM [72]	0.531
DSEBM [72]	0.610	ADT [72]	0.860
ADT-120 [73]	0.690	ADT-500 [73]	0.730
ADT-1000 [73]	0.750	COREL-120 [73]	0.760
AnoGAN [74]	0.615 ± 0.027	Deep-SVDD [74]	0.648 ± 0.020

Our results (cf. Table IX) show that SVM and RF are the best classifiers on the CIFAR-10 dataset. There is a swap on the first ranked classifier while using our proposed selection methods. However, the results are comparable. Moreover, our proposed methods outperform the state of the art methods regardless of the classifier used.

V. Conclusion

We introduced an approach that learns representations of feature descriptors, and then fuse them into a distribution-space. This probabilistic space allows us to learn a representation of the inputs' characteristics class, and, consequently, approximate what outliers may look like. Then, we synthesize outliers as drawings from the outskirts-distributions of the training samples. Finally, we demonstrated that using the given inputs and the synthesized outliers yields classifiers capable of detecting outliers consistently on several datasets with varying types and amounts of outliers. Moreover, we showed the importance of the different parts of our proposal: individual representation through compression, variational inference for fusion, and sampling for synthesis, through an ablation study.

As future work, we envision applications of our proposed idea beyond images, i.e., other domains like text, audio, etc.. We intend to explore other generative models for outlier synthesis to increase the robustness of our outliers. Another branch to be explored is to train the synthesis and the classifiers in a tightly coupled way, i.e., an end-to-end model from data representation, distillation, outlier synthesis, and classification, to compare if this approach yields better results than a loosely coupled one, like our proposal.

APPENDIX A
EXPERIMENTAL SETUP

MNIST [60], [61] contains 60 000 handwritten digits from '0' to '9.' Each of the ten digit categories is taken as the target class (i.e., inliers) and we simulate outliers (during testing) by randomly sampling images from other categories with a proportion of 10% to 50%. This experiment is repeated for all of the ten digit categories with the images being of size 28×28.

Caltech-256 [62] dataset contains 256 different object categories with a total number of 30 607 images. Each category has at least 80. For this dataset the images' size is 28×28 and we repeat same procedure three times by using images from $n \in \{1, 3, 5\}$ randomly chosen categories as inliers. From each category the first 150 images are used only if that category has more than 150 images. We then randomly select 50% outliers from the 'clutter' category which contains 827 images of different varieties in each experiment.

Coil-100 [63] dataset contains 100 different object categories with a total number of 7200 images. Each object category has 72 images taken from different pose intervals of 5 degrees and image's size is 32×32. We performed three different experiments by randomly selecting $n \in \{1, 4, 7\}$ categories images as inliers, and outliers were randomly chosen from other categories (at most one from each category) by varying percentage of outliers among 50%, 25%, and 15%.

Extended Yale B [64], [65] dataset contains frontal face images of 38 persons as categories. Each category has 64 different illumination conditions. The actual size of face images is 192×168, and for our experiment we down-sample them to 48×42. We performed two different experiments by randomly selecting $n \in \{1, 3\}$ categories images as inliers, and outliers are randomly chosen from other categories (at most one from each category) by varying the outliers percentage between 35% and 15%.

UCSD Pedestrian [66] dataset contains two subsets of Ped1 and Ped2 with a different street scenes recorded with a stationary camera at 10 fps and resolutions 158×234 and 240×360, respectively. The density of the pedestrians varies from low to high. The normal objects in all scenes are pedestrians (i.e., inliers). All other objects (e.g., cars, skateboarders, wheelchairs, etc.) are considered outliers. For the experiments the images size is down-sampled to 28×28. We performed different experiments on Ped1 scenes by randomly selecting normal object images from 34 categories as inliers, outlier images were randomly chosen from the remaining categories and each category contains 200 frames. For Ped2 similar experiments were performed but with 16 and 12 object categories as inliers and outlier, respectively, where the number of frames of each clip varies in outliers.

ISIC 2018 Challenge (Task 3) [67], [68] dataset contains seven different diseases as categories with a total number of 10015 images. The actual size of disease images is 450×600, and for our experiment we down sample them to 32×32 and try to implement deep architecture with limited resources. We performed six different experiments by considering NV disease category images as inliers and we tested outliers by select the 100 images from the rest of disease categories.

TABLE A.1. Amount of images on the ISIC 2018 Challenge (Task 3) Dataset.

Disease	No. of Images
MEL	1113
NV	6705
BCC	514
AKIEC	327
BKL	1099
DF	115
VASC	142

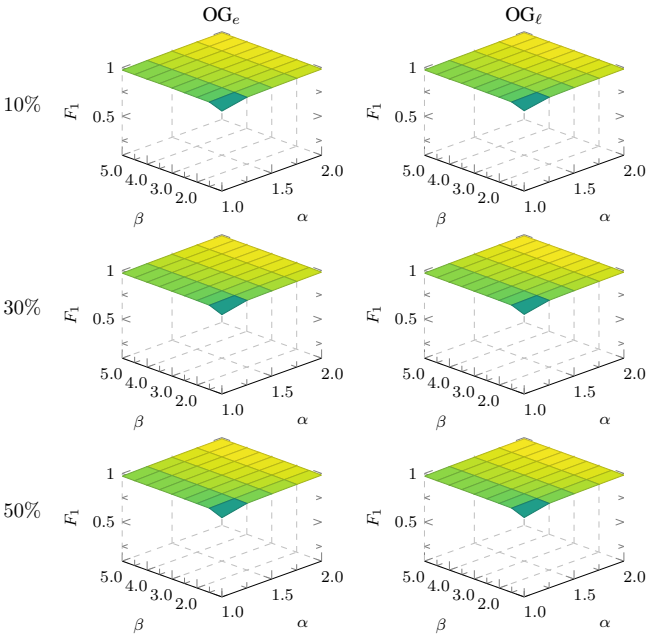


Fig. B.1. Comparisons of F_1 scores on MNIST dataset for the proposed methods by varying α and β parameters when inliers come from each class, and outliers are taken from three different percentages (%). We report average values among all ten classes of the digits.

CIFAR-10 [69] dataset consists of 60 000 32×32 color images in 10 classes, with 6000 images per class. There are 50 000 training images and 10 000 test images, divided equally across the classes. For the experiments, we use a single class for training, and the rest classes as outliers (similar to the previous protocols). We report averaged results overall classes with 5-fold cross-validation.

APPENDIX B
EXTENDED OUTLIER SYNTHESIS' PARAMETERS

We present high resolution versions for Fig. 3, and all configurations for all experiments in the six datasets (according to the setup shown in Appendix A) in Figs. B.1, B.2, B.3, B.4, B.5, and B.6.

APPENDIX C
EXTENDED RESULTS

We show the details of the true positive rate (TPR) and false positive rate (FPR) for the MNIST, Caltech-256, Coil-100, Extended Yale-B, UCSD Pedestrian, and ISIC 2018 Challenge (Task 3) datasets experiments in Tables C.1, C.2, C.3, C.4, C.5, and C.6, respectively.

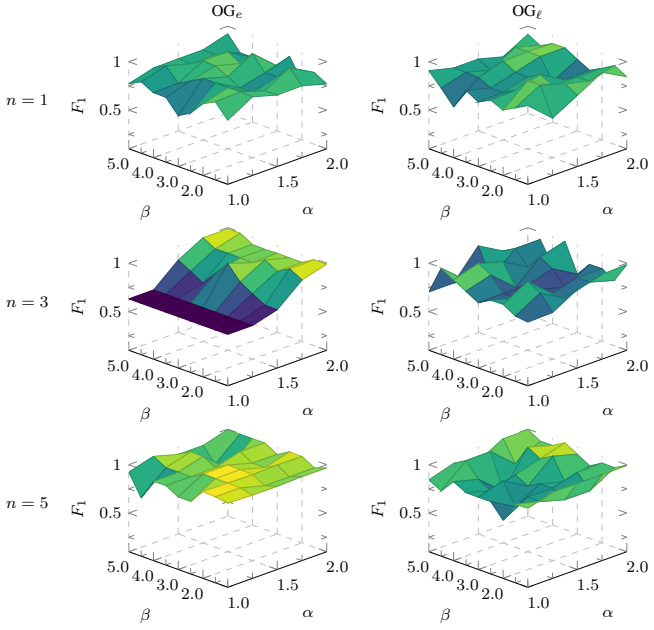


Fig. B.2. Comparisons of F_1 scores on Caltech-256 dataset for the proposed methods by varying α and β parameters when inliers are taken to be images from $n \in \{1, 3, 5\}$ randomly chosen categories with 50% outlier samples randomly selected from category clutter.

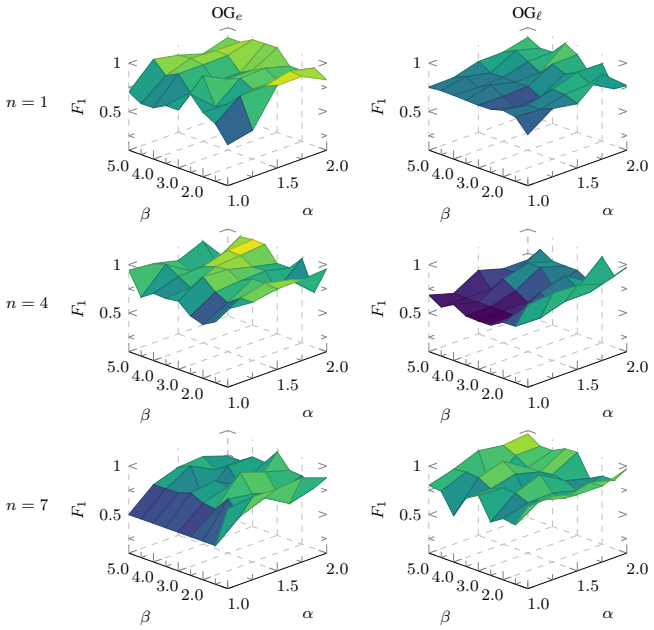


Fig. B.3. Comparisons of F_1 scores on Coil-100 dataset for the proposed methods by varying α and β parameters when inliers are taken to be images from $n \in \{1, 4, 7\}$ randomly chosen categories with 50%, 25%, and 15% outlier samples, respectively, randomly selected from the remaining categories.

Appendix D
Implementation Details

Our proposed framework comprises three main modules: deterministic and variational autoencoders (as shown in Fig. 2), and the classifier.

For the feature encoding, we use denoising autoencoders [84]. There are four hyperparameters that we used for each denoising

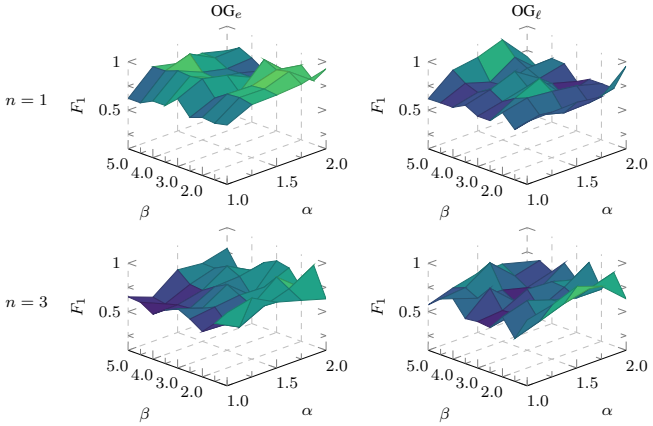


Fig. B.4. Comparisons of F_1 scores on Extended Yale B dataset for the proposed methods by varying α and β parameters when inliers are taken from $n \in \{1, 3\}$ randomly chosen subjects, respectively, and outliers are randomly chosen from the other subjects (at most one from each subject).

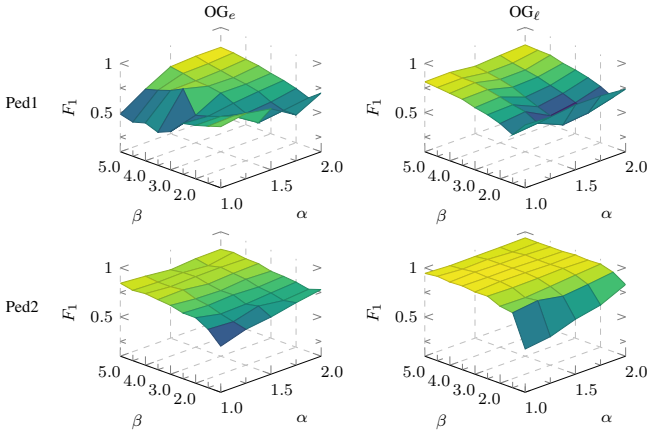


Fig. B.5. Comparisons of F_1 scores on UCSD Pedestrian dataset for the proposed methods by varying α and β parameters when inliers come from each class, and outliers are randomly selected from the remaining classes. We report the average values among all categories of video scenes.

autoencoder: number of layers, neurons per layer, type of activation function, and type of reconstruction loss. We evaluated an amount of layers among 1, 3, 5, and 7. For the number of neurons, we used three settings 'same,' 'double,' and 'half' number of neurons of the input per layer. The decoder is a mirror of the encoder in terms of layer structure. We evaluated four types of activation functions: ReLU, sigmoid, tanh, and linear. And for the reconstruction loss function (10), we used mean squared error (MSE) or binary cross entropy. If the input values are in $[0, 1]$, then we use cross entropy, otherwise we use MSE.

For the fusion, we use a VAE with a Gaussian distribution. There are three hyperparameters that we tuned for our VAE: number of layers, neurons per layer, and dimensionality of the Gaussian. Similarly to the AEs, we used between 1 and 2 layers, with 'same' and 'double' amount of neurons of the input per layer. And the sizes of the Gaussians vary from 2 to 6 (with step one)—we selected small dimensions due to resource constraints for executing the experiments.

There are three hyper-parameters that we tuned for our

TABLE C.1. Results of different percentage (%) of outliers on the MNIST dataset for the proposed methods: (a) OG_e, and (b) OG_ℓ. The top row show the zero to nine digit inlier categories an d average values among all ten classes of the digits.

(a)

	%	0	1	2	3	4	5	6	7	8	9	Avg.
TPR	10	0.984	0.984	0.984	0.983	0.985	0.982	0.983	0.982	0.986	0.983	0.984
FPR		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.004	0.001
TPR	20	0.986	0.982	0.985	0.982	0.986	0.979	0.986	0.982	0.986	0.983	0.984
FPR		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.004	0.001
TPR	30	0.985	0.984	0.984	0.983	0.984	0.982	0.984	0.981	0.986	0.982	0.984
FPR		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.004	0.001
TPR	40	0.983	0.983	0.984	0.983	0.984	0.982	0.984	0.980	0.986	0.983	0.983
FPR		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.004	0.001
TPR	50	0.984	0.984	0.984	0.982	0.984	0.981	0.983	0.980	0.986	0.984	0.983
FPR		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.004	0.001

(b)

	%	0	1	2	3	4	5	6	7	8	9	Avg.
TPR	10	0.986	0.984	0.984	0.986	0.986	0.984	0.985	0.981	0.986	0.983	0.984
FPR		0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000
TPR	20	0.985	0.983	0.984	0.983	0.984	0.983	0.983	0.980	0.987	0.983	0.983
FPR		0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000
TPR	30	0.984	0.984	0.985	0.983	0.983	0.982	0.984	0.981	0.985	0.982	0.983
FPR		0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000
TPR	40	0.983	0.983	0.984	0.983	0.984	0.982	0.983	0.981	0.985	0.982	0.983
FPR		0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000
TPR	50	0.985	0.983	0.984	0.983	0.984	0.981	0.984	0.981	0.986	0.981	0.983
FPR		0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000

TABLE C.2. Results of 50% randomly chosen outliers from category 257-clutter of the Caltech-256 dataset for the proposed methods: (a) OG_e, and (b) OG_ℓ. The top row shows the 2, 4, 6, 8, and 10 inlier categories and average values among all five classes of the objects.

(a)

	%	2	4	6	8	10	Avg.
TPR	50	0.976	0.894	0.901	0.899	0.971	0.928
FPR		0.000	0.092	0.084	0.086	0.000	0.052

(b)

	%	2	4	6	8	10	Avg.
TPR	50	0.979	0.894	0.896	0.898	0.971	0.927
FPR		0.000	0.087	0.084	0.087	0.094	0.070

SVMs: kernel type, kernel coefficient (γ), and regularization. We evaluated four types of kernels: linear, polynomial, RBF, and sigmoid. For the kernel coefficient, we used values 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, and 10^{-5}. And for the regularization parameter (C) we used 10^{-3}, 10^{-2}, 10^{-1}, 1, and 10.

APPENDIX E
HYPERPARAMETERS

A. Features

We show the hyperparameters we used for extracting the features, HOG [29] and LBP [59], in Tables E.1 and E.2.

B. Autoencoders (AEs), Variational Autoencoders (VAEs) and SVM

We show the best set of hyperparameters found on experimental results for MNIST, Caltech-256, Coil-100, Extended Yale-B, UCSD Pedestrian, and ISIC 2018 Challenge (Task 3) on Tables E.3, E.4, E.5, E.6, E.7, and E.8, respectively. The corresponding original results (on the paper) are in Tables I, II, III, IV, V, and VI.

TABLE C.3. Results of 50%, 25%, and 15% outliers of the Coil-100 dataset for the proposed methods: (a) OG_e, and (b) OG_ℓ. The top row shows the 2, 4, 6, 8, and 10 inlier categories and average values among all five classes of the objects.

(a)

	%	2	4	6	8	10	Avg.
TPR	50	0.933	0.928	0.934	0.922	0.949	0.933
FPR		0.069	0.075	0.068	0.080	0.040	0.066
TPR	25	0.959	0.972	0.955	0.940	0.964	0.958
FPR		0.041	0.021	0.040	0.059	0.029	0.038
TPR	15	0.877	0.860	0.870	0.884	0.856	0.869
FPR		0.118	0.126	0.109	0.114	0.139	0.121

(b)

	%	2	4	6	8	10	Avg.
TPR	50	0.903	0.893	0.888	0.892	0.924	0.900
FPR		0.086	0.093	0.102	0.090	0.074	0.089
TPR	25	0.900	0.903	0.968	0.890	0.924	0.917
FPR		0.069	0.072	0.039	0.092	0.072	0.068
TPR	15	0.837	0.850	0.917	0.876	0.860	0.868
FPR		0.154	0.102	0.073	0.118	0.129	0.115

TABLE C.4. Results of 35% and 15% outliers of the Extended Yale B dataset for the proposed methods: (a) OG_e, and (b) OG_ℓ. The top row shows the 2, 4, 6, 8, and 10 inlier categories and average values among all five classes of the objects.

(a)

	%	2	4	6	8	10	Avg.
TPR	35	0.941	0.870	0.844	0.921	0.913	0.897
FPR		0.049	0.075	0.083	0.076	0.074	0.061
TPR	15	0.882	0.860	0.862	0.910	0.876	0.869
FPR		0.048	0.079	0.088	0.059	0.083	0.070

(b)

	%	2	4	6	8	10	Avg.
TPR	35	0.932	0.852	0.837	0.901	0.923	0.889
FPR		0.037	0.092	0.088	0.081	0.074	0.067
TPR	15	0.899	0.907	0.872	0.908	0.891	0.882
FPR		0.053	0.049	0.052	0.069	0.068	0.062

REFERENCES

[1] M. Ahmed, A. N. Mahmood, and M. R. Islam, "A survey of anomaly detection techniques in financial domain," *Fut. Gen. Comput. Syst.*, vol. 55, pp. 278–288, 2016.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, p. 15, 2009.

[3] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artif. Intell. Rev.*, vol. 22, no. 2, pp. 85–126, 2004.

[4] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Process.*, vol. 99, pp. 215–249, 2014.

[5] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, vol. 2, pp. 60–65, IEEE, 2005.

[6] Y. Cong, J. Yuan, and J. Liu, "Sparse reconstruction cost for abnormal event detection," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 3449–3456, IEEE, 2011.

[7] W. Li, V. Mahadevan, and N. Vasconcelos, "Anomaly detection and localization in crowded scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 1, pp. 18–32, 2014.

TABLE C.5. Results of randomly chosen outliers of the UCSD Pedestrian dataset (on both sets Ped1 and Ped2) for the proposed methods: (a) OG_e, and (b) OG_ℓ. The top row shows the 1, 3, 5, 7, and 9 inlier categories and average values among all five classes of the scenes.

(a)

Subset		1	3	5	7	9	Avg.
Ped-1	TPR	0.936	0.946	0.951	0.939	0.972	0.949
	FPR	0.019	0.015	0.011	0.050	0.006	0.020
Ped-2	TPR	0.969	0.963	0.935	0.886	0.974	0.945
	FPR	0.006	0.017	0.006	0.097	0.015	0.029

(b)

Scene		1	3	5	7	9	Avg.
Ped-1	TPR	0.943	0.928	0.947	0.916	0.938	0.934
	FPR	0.022	0.021	0.049	0.013	0.005	0.022
Ped-2	TPR	0.961	0.985	0.962	0.988	0.923	0.964
	FPR	0.003	0.049	0.007	0.007	0.002	0.013

TABLE C.6. Results of randomly chosen outliers of the ISIC 2018 Challenge (Task 3) dataset for the proposed methods: (a) OG_e, and (b) OG_ℓ. The top row shows the disease categories for each outlier category, and average values among all.

(a)

	AKIEC	BCC	BKL	DF	MEL	VASC	All	Avg.
TPR	0.451	0.415	0.459	0.482	0.481	0.438	0.415	0.449
FPR	0.007	0.002	0.017	0.020	0.007	0.008	0.024	0.009

(b)

	AKIEC	BCC	BKL	DF	MEL	VASC	All	Avg.
TPR	0.636	0.676	0.879	0.845	0.666	0.703	0.745	0.736
FPR	0.013	0.020	0.011	0.017	0.016	0.018	0.020	0.016

TABLE E.1. HOG's hyperparameters. The best cased used in our proposed experiment is typeset in bold. Abbreviations: cells per block (CpB), orientations (O).

CpB	O	Pixels per Cell		
		(12 , 12)	(14 , 14)	(16 , 16)
(1 , 1)	9	(1,1) - 9 - (12,12)	**(1,1) - 9 - (14,14)**	(1,1) - 9 - (16,16)
	12	(1,1) - 12 - (12-12)	(1,1) - 12 - (14,14)	(1,1) - 12 - (16,16)
	15	(1,1) - 15 - (12,12)	(1,1) - 15 - (14,14)	(1,1) - 15 - (16,16)
(2 , 2)	9	(2,2) - 9 - (12,12)	(2,2) - 9 - (14,14)	(2,2) - 9 - (16,16)
	12	(2,2) - 12 - (12,12)	(2,2) - 12 - (14,14)	(2,2) - 12 - (16,16)
	15	(2,2) - 15 - (12,12)	(2,2) - 15 - (14,14)	(2,2) - 15 - (16,16)
(4 , 4)	9	(4,4) - 9 - (12,12)	(4,4) - 9 - (14,14)	(4,4) - 9 - (16,16)
	12	(4,4) - 12 - (12,12)	(4,4) - 12 - (14,14)	(4,4) - 12 - (16,16)
	15	(4,4) - 15 - (12,12)	(4,4) - 15 - (14,14)	(4,4) - 15 - (16,16)

TABLE E.2. LBP's hyperparameters. The best result is typeset in bold. Abbreviations: number of patterns (NP).

Method	# NP	Radius of circle		
		8	16	24
Default	4	Default-4-8	Default-4-16	Default-4-24
	6	Default-6-8	Default-6-16	Default-6-24
	8	Default-8-8	Default-8-16	Default-8-24
ROR	4	ROR-4-8	ROR-4-16	ROR-4-24
	6	ROR-6-8	ROR-6-16	ROR-6-24
	8	ROR-8-8	ROR-8-16	ROR-8-24
Uniform	4	**Uniform-4-8**	Uniform-4-16	Uniform-4-24
	6	Uniform-6-8	Uniform-6-16	Uniform-6-24
	8	Uniform-8-8	Uniform-8-16	Uniform-8-24
VAR	4	VAR-4-8	VAR-4-16	VAR-4-24
	6	VAR-6-8	VAR-6-16	VAR-6-24
	8	VAR-8-8	VAR-8-16	VAR-8-24

[8] M. Sabokrou, M. Fayyaz, M. Fathy, and R. Klette, "Deep-cascade: cascading 3D deep neural networks for fast anomaly detection and localization in crowded scenes," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 1992–2004, 2017.

[9] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli, "Adversarially learned one-class classifier for novelty detection," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 3379–3388, 2018.

[10] Y. Xia, X. Cao, F. Wen, G. Hua, and J. Sun, "Learning discriminative reconstructions for unsupervised outlier removal," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 1511–1519, 2015.

[11] C. You, D. P. Robinson, and R. Vidal, "Provable selfrepresentation based outlier detection in a union of subspaces," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 1–10, 2017.

[12] E. Eskin, "Anomaly detection over noisy data using learned probability distributions," in *Inter. Conf. Mach. Learn. (ICML)*, 2000.

[13] J. Kim and C. D. Scott, "Robust kernel density estimation," *J. Mach. Learn. Res.*, vol. 13, no. Sep, pp. 2529–2565, 2012.

[14] G. Lerman, M. B. McCoy, J. A. Tropp, and T. Zhang, "Robust computation of linear models by convex relaxation," *Found. Comput. Math.*, vol. 15, no. 2, pp. 363–410, 2015.

[15] M. Markou and S. Singh, "Novelty detection: a review—part 1: statistical approaches," *Signal Process.*, vol. 83, no. 12, pp. 2481–2497, 2003.

[16] K. Yamanishi, J.-I. Takeuchi, G. Williams, and P. Milne, "On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms," *Data Min. Knowl. Discov.*, vol. 8, no. 3, pp. 275–300, 2004.

[17] P. Bodesheim, A. Freytag, E. Rodner, M. Kemmler, and J. Denzler, "Kernel null space methods for novelty detection," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 3374–3381, 2013.

[18] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *ACM Conf. Manag. Data (ACM SIGMOD)*, vol. 29, pp. 93–104, ACM, 2000.

[19] J. Liu, Z. Lian, Y. Wang, and J. Xiao, "Incremental kernel null space discriminant analysis for novelty detection," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 792–800, 2017.

[20] M. Rahmani and G. K. Atia, "Coherence pursuit: Fast, simple, and robust principal component analysis," *IEEE Trans. Signal Process.*, vol. 65, no. 23, pp. 6260–6275, 2017.

[21] M. Sabokrou, M. Fathy, and M. Hoseini, "Video anomaly detection and localisation based on the sparsity and reconstruction error of auto-encoder," *Electron. Lett.*, vol. 52, no. 13, pp. 1122–1124, 2016.

[22] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 733–742, IEEE, 2016.

[23] M. Kimura and T. Yanagihara, "Semi-supervised anomaly detection using GANs for visual inspection in noisy training data," *arXiv:1807.01136*, 2018.

[24] S. Pidhorskyi, R. Almohsen, D. A. Adjeroh, and G. Doretto, "Generative probabilistic novelty detection with adversarial autoencoders," in *Adv. Neural Inf. Process. Sys. (NeurIPS)*, 2018.

[25] M. Ravanbakhsh, M. Nabi, E. Sangineto, L. Marcenaro, C. Regazzoni, and N. Sebe, "Abnormal event detection in videos using generative adversarial nets," in *IEEE Inter. Conf. Image Process. (ICIP)*, pp. 1577–1581, IEEE, 2017.

[26] M. Sabokrou, M. Fathy, M. Hoseini, and R. Klette, "Real-time anomaly detection and localization in crowded scenes," in *IEEE Inter. Conf. Comput. Vis., Pattern Recog. Wksps. (CVPRW)*, pp. 56–62, 2015.

[27] 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 *Inter. Conf. Inf. Process. Medical Imag. (IPMI)*, pp. 146–157, Springer, 2017.

[28] H.-g. Wang, X. Li, and T. Zhang, "Generative adversarial network based novelty detection usingminimized reconstruction error," *Front. Inf. Technol. Electron. Eng.*, vol. 19, no. 1, pp. 116–125, 2018.

[29] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, vol. 1, pp. 886–893, IEEE, 2005.

[30] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recogn.*, vol. 29, no. 1, pp. 51–59, 1996.

[31] D. Huang, C. Shan, M. Ardabilian, Y. Wang, and L. Chen, "Local binary patterns and its application to facial image analysis: a survey," *IEEE*

TABLE E.3. Best set of hyperparameters for MNIST dataset in proposed experiments. Abbreviations: Amount of Layers (AL), Number of neurons per layer (NN), Activation Function (AF), Latent Dimension (LD), Kernel coefficient (γ), and regularization parameter (C).

Method	AutoEncoder			Variational AutoEncoder			SVM		
	AL	NN	AF	LD	Alpha	Beta	Kernel	γ	C
OG_e	1	Same	ReLU	2	3.0	4.5	rbf	1	0.1
OG_ℓ	1	Same	ReLU	2	3.0	5.0	rbf	1	0.1

TABLE E.4. Best set of hyperparameters for Caltech-256 dataset in proposed experiments. Abbreviations: Amount of Layers (AL), Number of neurons per layer (NN), Activation Function (AF), Latent Dimension (LD), Kernel coefficient (γ) and regularization parameter (C).

Method	AutoEncoder			Variational AutoEncoder			SVM		
	AL	NN	AF	LD	Alpha	Beta	Kernel	γ	C
OG_e	5	Half	ReLU	2	1.75	3.0	rbf	1	0.1
OG_ℓ	5	Half	ReLU	2	1.25	2.0	rbf	1	0.1
OG_e	5	Same	ReLU	4	1.5	3.0	rbf	1	0.1
OG_ℓ	7	Same	ReLU	2	2.0	3.5	rbf	1	0.1
OG_e	5	Same	ReLU	2	1.0	2.0	rbf	1	0.1
OG_ℓ	7	Same	ReLU	4	1.5	4.5	rbf	1	0.1

TABLE E.5. Best set of hyperparameters for Coil-100 dataset in proposed experiments. Abbreviations: Amount of Layers(AL), Number of neurons per layer (NN), Activation Function (AF), Latent Dimension (LD), Kernel coefficient (γ) and regularization parameter (C).

Method	AutoEncoder			Variational AutoEncoder			SVM		
	AL	NN	AF	LD	Alpha	Beta	Kernel	γ	C
OG_e	5	Half	ReLU	2	2.0	3.5	rbf	1	0.1
OG_ℓ	1	Same	ReLU	2	1.75	1.0	rbf	1	0.1
OG_e	1	Double	ReLU	3	2.0	4.0	rbf	1	0.1
OG_ℓ	3	Same	ReLU	2	2.0	2.0	rbf	1	0.1
OG_e	1	Double	ReLU	3	1.5	2.5	rbf	1	0.1
OG_ℓ	3	Half	ReLU	2	1.5	5.0	rbf	1	0.1

TABLE E.6. Best set of hyperparameters for Extended Yale B dataset in proposed experiments. Abbreviations: Amount of Layers(AL), Number of neurons per layer(NN), Activation Function (AF), Latent Dimension (LD), Kernel coefficient (γ) and regularization parameter (C).

Method	AutoEncoder			Variational AutoEncoder			SVM		
	AL	NN	AF	LD	Alpha	Beta	Kernel	γ	C
OG_e	3	Half	ReLU	3	1.75	2.5	rbf	1	25
OG_ℓ	1	Half	ReLU	4	2.0	1.0	rbf	1	25
OG_e	5	Half	ReLU	3	2.0	1.5	rbf	1	0.1
OG_ℓ	5	Half	ReLU	3	1.5	1.0	rbf	1	0.1

Trans. Syst., Man, Cybern. C, vol. 41, no. 6, pp. 765–781, 2011.

[32] A. Ramírez Rivera, J. Rojas Castillo, and O. Chae, "Local directional number pattern for face analysis: Face and expression recognition," *IEEE Trans. Image Process.*, vol. 22, no. 5, pp. 1740–1752, 2013.

[33] A. Ramírez Rivera and O. Chae, "Spatiotemporal directional number transitional graph for dynamic texture recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 2146–2152, 2015.

[34] A. Ramírez Rivera, J. Rojas Castillo, and O. Chae, "Local directional texture pattern image descriptor," *Pattern Recogn. Lett.*, vol. 51, no. 0, pp. 94–100, 2015.

[35] F. G. Venhuizen, B. van Ginneken, B. Bloemen, M. J. van Grinsven, R. Philipsen, C. Hoyng, T. Theelen, and C. I. Sánchez, "Automated age-related macular degeneration classification in oct using unsupervised feature learning," in *Med. Imag. CAD*, vol. 9414, p. 94141I, International Society for Optics and Photonics, 2015.

[36] B. T. Morris and M. M. Trivedi, "Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2287–2301, 2011.

[37] 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," *Pattern Recogn.*, vol. 58, pp. 121–134, 2016.

[38] D. Xu, E. Ricci, Y. Yan, J. Song, and N. Sebe, "Learning deep representations of appearance and motion for anomalous event detection," *arXiv:1510.01553*, 2015.

[39] H. Xu, C. Caramanis, and S. Sanghavi, "Robust PCA via outlier pursuit," in *Adv. Neural Inf. Process. Sys. (NeurIPS)*, pp. 2496–2504, 2010.

[40] J. Park, D. H. Choi, Y.-B. Jeon, Y. Nam, M. Hong, and D.-S. Park, "Network anomaly detection based on probabilistic analysis," *Soft Computing*, vol. 22, no. 20, pp. 6621–6627, 2018.

[41] B. G. Atli, Y. Miche, A. Kalliola, I. Oliver, S. Holtmanns, and A. Lendasse, "Anomaly-based intrusion detection using extreme learning machine and aggregation of network traffic statistics in probability space," *Cog. Comput.*, vol. 10, no. 5, pp. 848–863, 2018.

[42] V. L. Cao, M. Nicolau, and J. McDermott, "One-class classification for anomaly detection with kernel density estimation and genetic programming," in *European Conference on Genetic Programming* (M. I. Heywood, J. McDermott, M. Castelli, E. Costa, and K. Sim, eds.), (Cham), pp. 3–18, Springer International Publishing, 2016.

[43] C. D. Scott and R. D. Nowak, "Learning minimum volume sets," *J. Mach. Learn. Res.*, vol. 7, no. Apr., pp. 665–704, 2006.

[44] A. O. Hero, "Geometric entropy minimization (GEM) for anomaly detection and localization," in *Adv. Neural Inf. Process. Sys. (NeurIPS)*, pp. 585–592, 2007.

[45] K. Sricharan and A. O. Hero, "Efficient anomaly detection using bipartite k-NN graphs," in *Adv. Neural Inf. Process. Sys. (NeurIPS)* (J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, eds.), pp. 478–486, Curran Associates, Inc., 2011.

[46] E. Hou, K. Sricharan, and A. O. Hero, "Latent laplacian maximum entropy discrimination for detection of high-utility anomalies," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 6, pp. 1446–1459, 2018.

[47] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Adv. Neural Inf. Process. Sys. (NeurIPS)*, pp. 582–588, 2000.

[48] S. K. Lim, Y. Loo, N.-T. Tran, N.-M. Cheung, G. Roig, and Y. Elovici, "Doping: Generative data augmentation for unsupervised anomaly detection with gan," in *IEEE Inter. Conf. Data Min. (ICDM)*, pp. 1122–1127, IEEE, 2018.

[49] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, "Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic," *IEEE Sensors Lett.*, vol. 3, no. 1, pp. 1–4, 2019.

[50] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Adv. Neural Inf. Process. Sys. (NeurIPS)*, pp. 2672–2680, 2014.

[51] J. Arias Figueroa and A. Ramírez Rivera, "Learning to cluster with auxiliary tasks: A semi-supervised approach," in *Conf. Graphics Patterns Images (SIBGRAPI)*, pp. 1–8, Oct. 2017.

[52] J. Arias Figueroa and A. Ramírez Rivera, "Is simple better?: Revisiting simple generative models for unsupervised clustering," in *Wksp. Bayesian Deep Learn. (NeurIPS)*, Dec. 2017.

[53] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Inter. Conf. Learn. Represent. (ICLR)*, 2014.

[54] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab, "Deep autoencoding models for unsupervised anomaly segmentation in brain MR images," in *Inter. MICCAI Brainlesion Wksp.*, pp. 161–169, Springer, 2018.

[55] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *WWW Conference on World Wide Web*, pp. 187–196, International World Wide Web Conferences Steering Committee, 2018.

[56] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, "Deep structured energy based models for anomaly detection," *arXiv:1605.07717*, 2016.

[57] M.-N. Nguyen and N. A. Vien, "Scalable and interpretable one-class SVMs with deep learning and random fourier features," in *European Conf. Princ. Data Min. Knowl. Discov. (ECML-PKDD)*, pp. 157–172, Springer, 2018.

[58] T. C. Landgrebe, D. M. Tax, P. Paclík, and R. P. Duin, "The interaction between classification and reject performance for distance-based reject-

TABLE E.7. Best set of hyperparameters for UCSD Pedestrian dataset in proposed experiments. Abbreviations: Amount of Layers(AL), Number of neurons per layer (NN), Activation Function (AF), Latent Dimension (LD), Kernel coefficient (γ) and regularization parameter (C).

Method	AutoEncoder			Variational AutoEncoder			SVM		
	AL	NN	AF	LD	Alpha	Beta	Kernel	γ	C
OG_e	5	Half	1	2	2	2.5	rbf	1	10
OG_ℓ	3	Half	1	2	2	2.5	rbf	1	10
OG_e	1	Half	1	3	1.25	2.0	rbf	1	10
OG_ℓ	5	Half	1	3	1.75	4.0	rbf	1	10

TABLE E.8. Best set of hyperparameters for ISIC 2018 Challenge (Task 3) dataset in proposed experiments. Abbreviations: Amount of Layers (AL), Number of neurons per layer (NN), Activation Function (AF), Latent Dimension (LD), Kernel coefficient (γ) and regularization parameter (C).

Method	AutoEncoder			Variational AutoEncoder			SVM		
	AL	NN	AF	LD	Alpha	Beta	Kernel	γ	C
OG_e	5	Half	1	12	1.25	1.0	rbf	1	1
OG_ℓ	3	Half	1	16	1.5	1.0	rbf	1	0.1
OG_e	5	Half	1	11	1.5	1.0	rbf	1	1
OG_ℓ	3	Half	1	14	1.0	1.0	rbf	1	1
OG_e	3	Half	1	10	1.0	1.0	rbf	1	0.1
OG_ℓ	3	Half	1	15	1.0	1.0	rbf	1	1
OG_e	5	Half	1	16	1.0	1.0	rbf	1	1
OG_ℓ	3	Half	1	14	1.0	1.0	rbf	1	1
OG_e	5	Half	1	17	1.0	1.0	rbf	1	1
OG_ℓ	3	Half	1	17	1.0	1.5	rbf	1	1
OG_e	3	Half	1	12	1.0	1.0	rbf	1	1
OG_ℓ	3	Half	1	18	1.5	1.0	rbf	0.1	0.1
OG_e	5	Half	1	17	1.0	1.0	rbf	1	0.1
OG_ℓ	3	Half	1	16	1.0	1.0	rbf	1	1

option classifiers," *Pattern Recogn. Lett.*, vol. 27, no. 8, pp. 908–917, 2006.

[59] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 12, pp. 2037–2041, 2006.

[60] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proc. IEEE*, vol. 86, pp. 2278–2324, 1998.

[61] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database." AT&T Labs, 2010.

[62] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," Tech. Rep. CaltechAUTHORS:CNS-TR-2007-001, Caltech, 2007.

[63] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (COIL100)," Tech. Rep. CUCS-006-96,, Columbia University, 1996.

[64] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 643–660, 2001.

[65] K. Lee, J. Ho, and D. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, pp. 684–698, 2005.

[66] A. Chan and N. Vasconcelos, "UCSD pedestrian dataset," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 909–926, 2008.

[67] P. Tschandl, C. Rosendahl, and H. Kittler, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific data*, vol. 5, p. 180161, 2018.

[68] N. C. Codella, D. Gutman, M. E. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, A. Kalloo, K. Liopyris, N. Mishra, H. Kittler, *et al.*, "Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (ISBI), hosted by the international skin imaging collaboration (ISIC)," in *IEEE Inter. Symp. Biomed. Imag. (ISBI)*, pp. 168–172, IEEE, 2018.

[69] A. Krizhevsky, "Learning multiple layers of features from tiny images," Master's thesis, University of Toronto, 2009.

[70] R. Chaker, Z. Al Aghbari, and I. N. Junejo, "Social network model for crowd anomaly detection and localization," *Pattern Recogn.*, vol. 61, pp. 266–281, 2017.

[71] Y. Lu and P. Xu, "Anomaly detection for skin disease images using variational autoencoder," *arXiv:1807.01349*, 2018.

[72] I. Golan and R. El-Yaniv, "Deep anomaly detection using geometric transformations," in *Adv. Neural Inf. Process. Sys. (NeurIPS)*, pp. 9758–9769, 2018.

[73] C. Hofer, R. Kwitt, M. Dixit, and M. Niethammer, "Connectivity-optimized representation learning via persistent homology," in *Inter. Conf. Mach. Learn. (ICML)*, 2019.

[74] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Inter. Conf. Mach. Learn. (ICML)*, pp. 4393–4402, 2018.

[75] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Inter. Conf. Mach. Learn. (ICML)*, pp. 663–670, 2010.

[76] M. C. Tsakiris and R. Vidal, "Dual principal component pursuit," in *IEEE Inter. Conf. Comput. Vis. Wksps. (ICCVW)*, pp. 10–18, 2015.

[77] M. Soltanolkotabi, E. J. Candes, *et al.*, "A geometric analysis of subspace clustering with outliers," *Anals Stats.*, vol. 40, no. 4, pp. 2195–2238, 2012.

[78] A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz, "Robust real-time unusual event detection using multiple fixed-location monitors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 3, pp. 555–560, 2008.

[79] R. Mehran, A. Oyama, and M. Shah, "Abnormal crowd behavior detection using social force model," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 935–942, IEEE, 2009.

[80] J. Kim and K. Grauman, "Observe locally, infer globally: a space-time mrf for detecting abnormal activities with incremental updates," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 2921–2928, IEEE, 2009.

[81] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly detection in crowded scenes," in *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 1975–1981, IEEE, 2010.

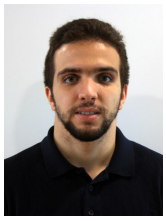
[82] J. Sun, X. Wang, N. Xiong, and J. Shao, "Learning sparse representation with variational auto-encoder for anomaly detection," *IEEE Access*, vol. 6, pp. 33353–33361, 2018.

[83] Y. Fan, G. Wen, D. Li, S. Qiu, and M. D. Levine, "Video anomaly detection and localization via gaussian mixture fully convolutional variational autoencoder," *arXiv:1805.11223*, 2018.

[84] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Inter. Conf. Mach. Learn. (ICML)*, pp. 1096–1103, ACM, 2008.

Adín Ramírez Rivera (S'12, M'14) received his B.Eng. degree in Computer Engineering from Universidad de San Carlos de Guatemala (USAC), Guatemala in 2009. He completed his M.Sc. and Ph.D. degrees in Computer Engineering from Kyung Hee University, South Korea in 2013. He is currently Assistant Professor at the Institute of Computing, University of Campinas, Brazil. His research interests are video understanding (including video classification, semantic segmentation, spatiotemporal feature modeling, and generation), and understanding and creating complex feature spaces.

Adil Khan received his B.S. degree in Information Technology from National University of Sciences and Technology (NUST), Pakistan in 2005. He completed his M.Sc. and Ph.D. degrees in Computer Engineering from Kyung Hee University, South Korea in 2011. He is currently Professor at the Institute of Artificial Intelligence and Data Science, Innopolis University, Russia. His research interests are machine learning and deep learning.



Imad Eddine Ibrahim Bekkouch received his B.S. degree in Computer science from Abdel Hamid Mehri Constantine 2 University, Algeria in 2018. He got his M.Sc. in data science at Innopolis University, Russia. Currently, he is pursuing his PhD in Sorbonne Center for Artificial Intelligence, Paris, France, and working as a research assistant at the Institute of Artificial Intelligence and Data Science, Innopolis University, Russia. His research interests are Domain adaptation, Computer Vision, Machine Learning and Deep Learning.

Taimoor Shakeel Sheikh received the B.Eng. (Telecommunication) and M.Eng. (Electrical) degrees from National University of Computer and Emerging Sciences (NUCES), Pakistan in 2012 and 2015 respectively. He completed his another M.Sc. degree in Informatics and Computer Science from Innopolis University,Russia, in 2019. He is currently an researcher at the Institute of Artificial Intelligence and Data Science, Innopolis University, Russia. His research interests are computer vision related tasks, and deep learning.

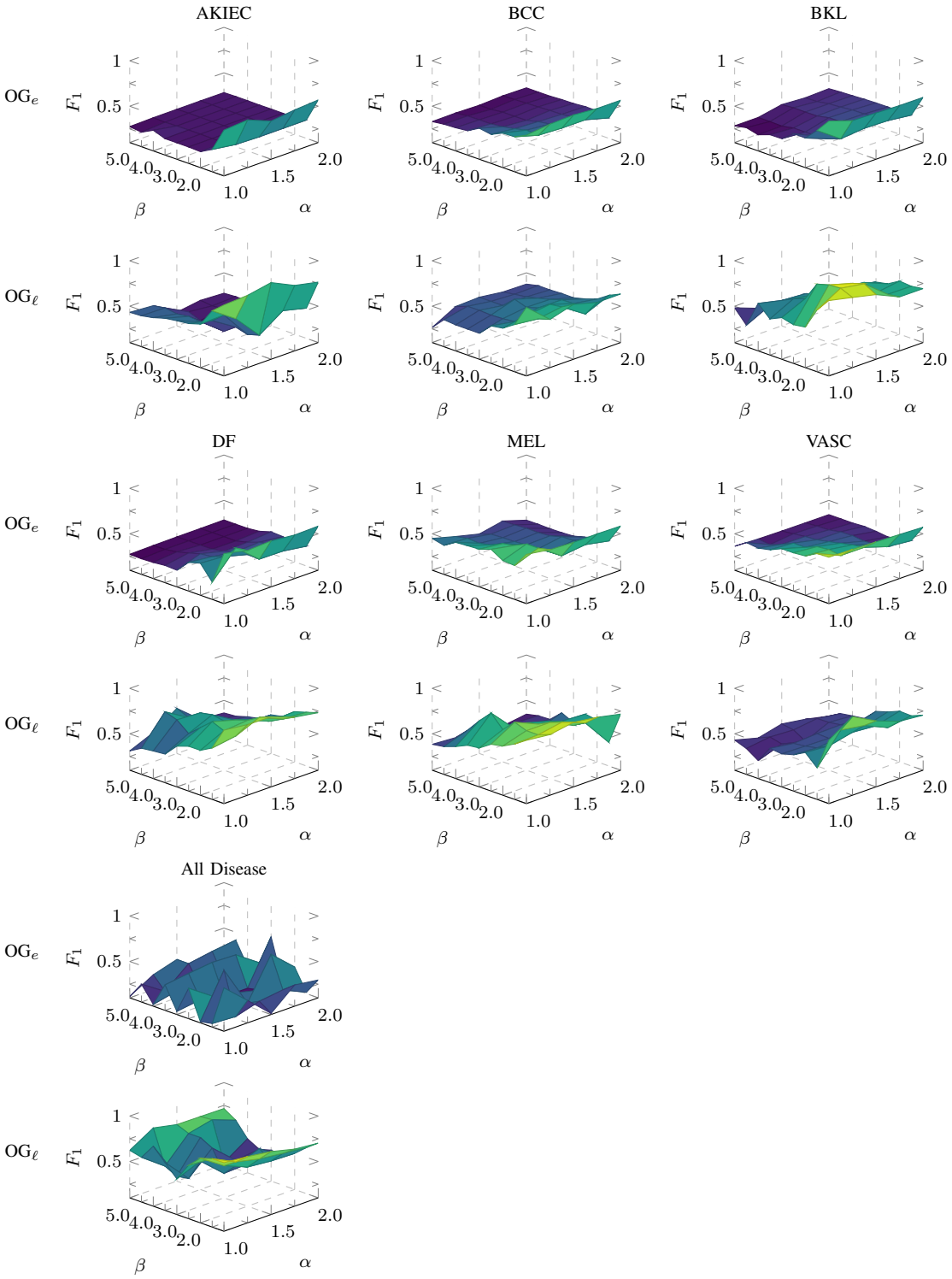


Fig. B.6. Comparisons of F_1 scores on ISIC 2018 Challenge (Task 3) dataset for the proposed methods by varying α and β parameters when inlier are taken images from NV disease category and outliers are randomly chosen from each individual disease category (show column wise, every two rows). Last rows show the experimental results when we consider random outliers from all disease categories together.