GridNet: Image-Agnostic Conditional Anomaly Detection for Indoor Surveillance

Ilker Bozcan, Jonas Le Fevre, Huy X. Pham, and Erdal Kayacan

***Abstract*—We present a deep autoencoder-based anomaly detection method (GridNet) for indoor surveillance. Unlike similar studies, GridNet is image-agnostic by taking a specific representation of a scene as inputs instead of the raw image itself. Its input is grid representations of scene images, which indicate spatial layouts of objects in a scene. This approach allows us to isolate the anomaly detection problem from any vision-related issues, such as illumination variations. In addition to grid representations, GridNet takes a location vector of a scene as input to learn the normalities of each scene conditioned on its location. We also propose a novel loss function that increases the model's reconstruction capability for grid representations. It enables the network to increase its precision and recall throughout the reconstruction. In our experiments, we compare our method with the existing studies on simulated and real-world data. The experimental results show the superiority of our method compared to the baseline methods. The code, data, and simulation environments will be available at https://bozcani.github.io/gridnet.**

***Index Terms*—Anomaly detection, autonomous surveillance, deep learning in robotics, unmanned autonomous vehicles, robot learning.**

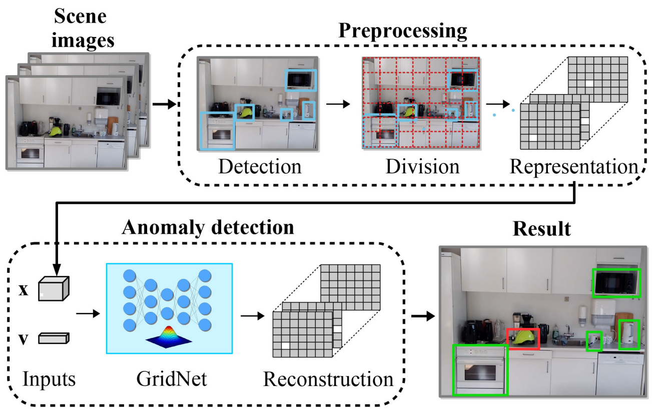


Fig. 1. Schematic representation of the proposed anomaly detection framework. Extracting the grid representation of the detected objects is the preliminary step. Then, GridNet takes the scene representation ($\mathbf{x}$) and the location vector ($\mathbf{v}$) as inputs, and it tries to reconstruct the original input. The differences between the input and the reconstruction indicate anomalies in the scene. [Best viewed in color].

I. Introduction

Anomaly detection is the task of identifying abnormal observations that significantly deviate from the majority of data [1]. However, it is not a straightforward machine learning problem due to several reasons. Firstly, anomalies are -by definition- rare compared to regular observations. Therefore, lack of labeled data is a common issue in anomaly detection problems. Secondly, an observation may be considered normal or abnormal, depending on the context that the observation is made. Hence, anomaly detection is conditional reasoning instead of making an objective decision. Lastly, in the scope of visual surveillance, anomaly detection might be considered a holistic task, including visual perception and outlier detection. However, such a coupling between these two tasks might cause anomaly detection to fail due to vision-related problems such as illumination variations. This issue is valid for even indoor environments, where lighting could affect the appearance of a scene. As a result, an anomaly detection method for visual surveillance should be robust for vision-related problems and trained in an unsupervised manner to learn conditional anomalies.

In this work, we propose a deep autoencoder-based method (GridNet) for image-agnostic conditional anomaly detection. Our method can find object-wise anomalies that are out-of-context in scene images. Instead of a raw image, it takes a meta-representation of a scene image as an input (See Fig. 1). As meta-representation, we use a grid representation of a scene that we have proposed in our previous work [2]. The network does not see image features during the training and inference (i.e., image-agnostic). The separation of visual perception and anomaly detection tasks allows us to optimize the network to learn the normality of an environment on grid representations without concerning vision-related problems. GridNet takes a grid representation and a location vector for a scene, where the location vector holds a global position of the scene. The network is trained with only normal samples that are grid representations of scenes, which do not include anomalies (i.e., objects that are out of the context of a scene). The joint training on grid representations and location vectors allows the network to learn the normal distribution of grid representations conditioned on location vectors. Therefore, the network can infer anomalies for different scenes, even if their grid representations are similar. We

Manuscript received October 14, 2020; accepted January 21, 2021. Date of publication February 3, 2021; date of current version March 2, 2021. This letter was recommended for publication by Associate Editor D. Held and Editor D. Kulic upon evaluation of the Reviewers' comments. This work was supported by Aarhus University, Department of Electrical, and Computer Engineering (28173). *(Corresponding author: Ilker Bozcan.)*

The authors are with the Artificial Intelligence in Robotics Laboratory (Air Lab), Department of Electrical and Computer Engineering, Aarhus University, 8000 Aarhus C, Denmark (e-mail: ilker@ece.au.dk; jonas.le.fevre@ece.au.dk; huy.pham@ece.au.dk; erdal@ece.au.dk).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3057003, provided by the authors.
Digital Object Identifier 10.1109/LRA.2021.3057003

also propose a multi-part loss function (grid loss) to increase the model's reconstruction capability for grid representations. Grid loss is composed of (i) an adjustable variational loss term that optimizes the variational lower bound of the model [3], and (ii) a novel loss term (i.e., soft-f1 loss) that increases the harmonic mean of the precision and recall of the network for the reconstruction.

We compare the performance of GridNet with several baselines commonly used in anomaly detection on both simulated and real-world data. The experiments show the superiority of our method compared to the baselines.

A. Related Work

There are different approaches for anomaly detection according to their training schemes, including supervised, semi-supervised, and unsupervised [1]. We review unsupervised anomaly detection methods trained with a dataset containing only normal (i.e., anomaly-free) data. After the training, these methods can find anomalies that do not conform well with the normality.

We can divide unsupervised anomaly detection approaches for visual surveillance into several groups based on how the representations of normality are learnt [4]. Clustering-based ([5]), distance-based ([6]), and one-class support vector machine (SVM) methods ([7], [8]) learn the latent representation of normal data optimizing a particular anomaly measure. They are vulnerable to weak design of the anomaly measures since they are dependent on the measures. Autoencoders ([2], [9], [10]), sparse-coding ([11], [12]), generative adversarial networks ([13], [14]), and probabilistic models ([15]–[18]) learn the normality of training data optimizing a generic objective function, which is not necessarily designed for the anomaly detection task. However, they can still capture the underlying regularities of normal data, and differentiate outliers from normal samples.

The majority of existing anomaly detection methods for video surveillance take a raw image or a frame sequence as input. They are trained in end-to-end manner taking a raw input, and predicting anomalies in their output. Therefore, they are affected by perturbations of input images' visual appearance. Moreover, after training on a dataset, they may not work on another dataset if samples in both datasets are contextually similar, yet they differ in visual appearance. We set [2] and [19] apart from this group since they separate anomaly detection into two tasks: (i) environment perception and (ii) outlier detection. Bozcan *et al.* [2] propose a variant of variational autoencoders (namely, UAV-ADNet) to find object-wise anomalies for traffic surveillance scenarios [20]. Unlike similar studies related to anomaly detection in visual data ([7], [9]), they detect object-wise anomalies instead of pixel-level. Therefore, their method can represent anomalies explicitly. Hinami *et al.* [19] propose a framework consisting of (i) multi-task Fast RCNN to learn generic knowledge of an environment, and (ii) a statistical model learning the underlying distribution of generic knowledge.

B. Contributions

The contributions of this letter are:

- GridNet: We propose a variant of convolutional variational autoencoder that can learn the conditional distribution of object layouts for a given environment. Unlike existing studies, our approach is image-agnostic and separates perception and anomaly detection problems using meta representations of images as input. Therefore, GridNet is not affected by vision-related issues, and these issues are considered perception problems.
- Grid Loss: We propose a novel loss term to increase the network's reconstruction performance. The overall loss function is a weighted sum of the variational loss and a novel soft-f1 loss defined on grid representations. The proposed loss function increases the exactness (i.e., precision) and completeness (i.e., recall) of reconstructed samples.
- The Dataset: We propose three datasets for object-level anomaly detection collected in real and simulated environments. The datasets include samples consisting of scene images, object bounding box annotations, and scene location labels.

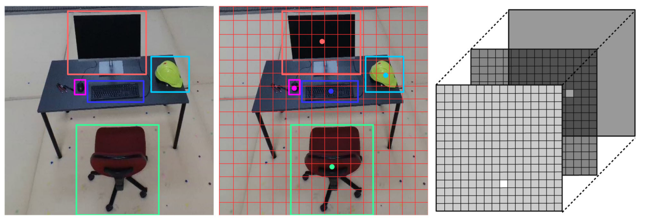
The remainder of the letter is organized as follows. Section II introduces the proposed network and the loss function. Section III presents the evaluation of the proposed method on a wide collection of simulated and real-world data, including discussions on the results. Section IV discusses the weaknesses and potentials of this work. Section V summarizes the work with some conclusions.

II. GRIDNET

We design a neural network (GridNet) that optimizes a custom-designed objective function (grid loss). The network takes the grid representation and location vector of a scene as inputs and learns spatial layouts of objects in a scene conditioned on a location vector. The motivation for using a reconstruction-based anomaly detection method is that the network is enforced to learn the regularities of the normal data during the training. Therefore, anomalies are difficult to be reconstructed from the learned representations and have large reconstruction errors. For inference, the network reconstructs the given input using its parameters. The difference between the original input and the reconstructed sample indicates anomalies in an environment.

A. Grid Representation of Scene Images

In our previous work [2], we propose a grid-based representation of an environment for aerial images captured by an unmanned aerial vehicle (UAV). In this representation, a grid cell corresponding to a specific image area has a value of 1 if an object belonging to a particular category is presented in this specific cell, otherwise 0 (See Fig. 2). The grid representation effectively reflects the spatial layouts of objects in the image plane. In this work, we use grid-wise representations of scenes as the input of the anomaly detection network. Therefore, the network uses a meta-representation of scenes instead of raw images, and it is abstracted from vision-related problems such as illumination.



(a) Raw image (b) Division of the image into grid regions (c) Binary 3D tensor for grid representation

Fig. 2. This figure illustrates the process of grid representation extraction proposed in [2]. Objects are detected in a raw image (a). Then, the scene is split into predefined grids (b). For each object category in the dataset, a grid matrix is created, representing the spatial layout of objects (c). A grid cell has a value of 1 if an object bounding box center lies in a particular grid area of the image.

B. The Network Architecture of GridNet

We implement GridNet as a hybrid architecture of convolutional and fully connected layers with residual connections (See Fig. 3). GridNet has a form of a variational autoencoder (VAE) [3] which takes high dimensional input data and compresses it into a smaller latent representation. Unlike a standard autoencoder, which learns a compressed representation of an input, a VAE learns the parameters of a probability distribution representing the data, such as the mean and variance of a Gaussian.

The network takes the grid representation and the location vector of a scene as inputs and reconstructs grid representation as output. To learn the latent distribution of training scenes, we use a sampling layer as in VAE [3]. After inference, a reconstruction of a test scene corresponds to its counterpart in the normal (i.e., training) samples. Therefore, anomalies should be removed in the reconstructed sample.

The network has three parts: (i) encoder, (ii) sampler, and (iii) decoder. The encoder ($\mathcal{E}$) accepts the grid representation and the location vector of a scene and finds a set of the parameter vectors (i.e., $\mathbf{z}_\mu$, $\mathbf{z}_\sigma$) for specifying the conditional distribution of the latent representation ($\mathbf{z}$). The encoder is formulated as follows:

$$\mathcal{E} : e_\psi(\mathbf{x}, \mathbf{v}) \rightarrow \{\mathbf{z}_\mu, \mathbf{z}_\sigma\}, \tag{1}$$

where e is a neural network with parameters ψ, and $\mathbf{x}$ and $\mathbf{v}$ are an input grid vector and a location vector, respectively. $\mathbf{z}_\mu$ and $\mathbf{z}_\sigma$ are the encoder's outputs corresponding to the mean and variance, respectively, of the latent distribution.

The sampler ($\mathcal{S}$) takes the parameters $\mathbf{z}_\mu$ and $\mathbf{z}_\sigma$, which are outputs of the encoder, as inputs, and it samples the vector $\mathbf{z}$ from the latent distribution defined by the parameters. This step is not straightforward since the random sampling operation is not differentiable. Therefore, it does not allow back-propagation during the training phase. To solve this problem, we re-write the sampler (S) using the re-parameterization trick as follows [21]:

$$\mathcal{S} \ : \ \mathbf{z}_\mu + \mathbf{z}_\sigma * \boldsymbol{\epsilon} \rightarrow \mathbf{z}, \tag{2}$$

$\boldsymbol{\epsilon}$ is a random noise that is generated from a standard normal distribution. It imitates the stochasticity in the random sampling. As a result, the sampler ($\mathcal{S}$) can be seen as a simple adding function of $\mathbf{z}_\mu$, $\mathbf{z}_\sigma$, and $\boldsymbol{\epsilon}$ that allows back-propagation.

The decoder ($\mathcal{D}$) takes the sampled vector $\mathbf{z}$ that is calculated by the sampler as input. Moreover, residual connections from the encoder's convolution layers and the location vector are fed to the network. As a result, the decoder tries to reconstruct the original input. The decoder can be formulated as follows:

$$\mathcal{D} : d_\omega(\mathcal{F}(\mathbf{x}), \mathbf{z}, \mathbf{v}) \rightarrow \hat{\mathbf{x}}, \tag{3}$$

where d is a neural network with the parameters ω. $\mathcal{F}(\mathbf{x})$ represents the activations calculated in $\mathcal{E}$ for the residual connections. $\hat{\mathbf{x}}$ is the reconstructed sample, which is the output of the network. The residual connections allow gradients to flow through the encoder without passing FC layers. We empirically set the residual connections. ReLU activation functions are applied after each convolutional and fully-connected layers, except the last fully-connected layer, where the linear activation function is applied. We do not use any pooling layer in our architecture to prevent information loss.

C. Grid Loss

We design a novel loss function (*grid loss*) to optimize the network. The loss function should constrain the network to (i) learn what normality for a scene is, and (ii) have high reconstruction capability. We compare the original input ($\mathbf{x}$) and the reconstructed sample ($\hat{\mathbf{x}}$) to find anomalies. If an object appears in a specific region in the image, then the corresponding grid cell has a value of 1 (i.e., active cell) otherwise 0 (i.e., passive cell). During inference, the grid cells for anomaly objects should be 0 in the reconstructed sample, whereas it is 1 in the original input. Moreover, the network should not remove normal objects in the reconstructed sample. Therefore, network should have high precision and recall in the reconstruction task. This task is not trivial since a grid representation is a sparse matrix dominated by zeros. Therefore, we introduce grid loss that enables the network to learn the training data distribution and have a high precision and high recall scores for the reconstruction of grid representations.

We introduce grid loss starting from the standard the evidence lower bound (ELBO) given in [3] with an additional regularizer term ($\mathcal{L}_{soft-f1}$):

$$\mathcal{L}_{\text{grid}} = \underbrace{\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}}}_{\text{ELBO}} + \mathcal{L}_{\text{soft-f1}}, \tag{4}$$

where $\mathcal{L}_{\text{rec}}$ is the reconstruction error penalizing the network for having the reconstructed samples that differ from the original input grids, and $\mathcal{L}_{\text{KL}}$ is a regularizer term that forces the network to have a constructed latent space. Note that $\mathcal{L}_{\text{rec}}$ and $\mathcal{L}_{\text{KL}}$ are slight modifications of the original VAE loss [3]. We propose a novel regularizer term, $\mathcal{L}_{\text{soft-f1}}$, to increase f1-score (i.e., the harmonic mean of precision and recall) for the model's reconstruction of grid representations. We explain all three terms ($\mathcal{L}_{\text{KL}}$, $\mathcal{L}_{\text{rec}}$, $\mathcal{L}_{\text{soft-f1}}$) as following subsections:

1) Regularizer for Latent Space ($\mathcal{L}_{KL}$): Using the definitions in Def. 1, 2, 3; the regularizer term given in [3] can be re-written

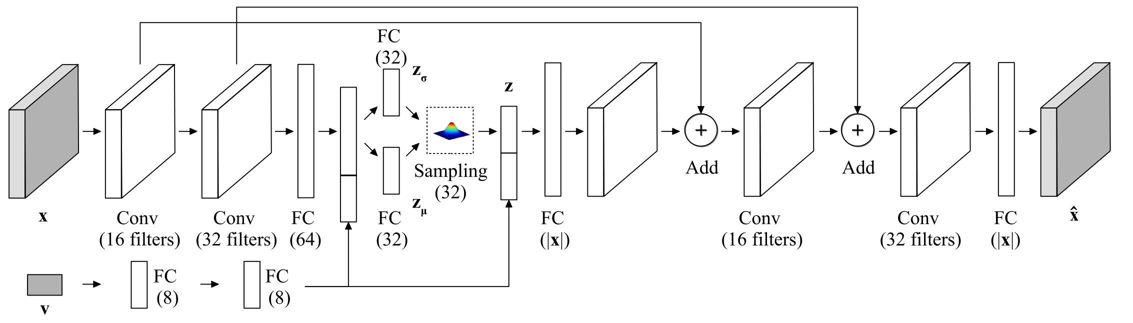


Fig. 3. Network architecture of GridNet. Grid representations ($\mathbf{x}$) and the location vector ($\mathbf{v}$) have different streams in the encoder part, and they are concatenated at the end of the encoder. The location vector is also fed to the beginning of the decoder to decode the grid representation conditioned on the location vector. The size of the last FC layer equals to the total number of cells in the grid representation ($|\mathbf{x}|$). At the end of the decoder, the FC layer is reshaped to the input grid's size.

for GridNet as follows:

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}\left[e_{\psi}(\mathbf{x}, \mathbf{v}) || \mathcal{N}(\mathbf{0}, \mathbf{1})\right]. \tag{5}$$

Note that (5) is Kullback-Leibler divergence of a unit Gaussian distribution ($\mathcal{N}(\mathbf{0}, \mathbf{1})$) with respect to the posterior distribution represented by the encoder. The constraint $\mathcal{L}_{\text{KL}}$ can be re-formulated using hidden nodes in the mean and the standard deviation vectors ($\mathbf{z}_{\mu}$ and $\mathbf{z}_{\sigma}$, respectively) which are the outputs of the encoder [21]:

$$\mathcal{L}_{\text{KL}} = \frac{1}{2}\left[\sum_{m=0}^{d}\left({z_{\sigma m}}^2 + {z_{\mu_m}}^2 - \log\left({z_{\sigma m}}^2\right) - 1\right)\right], \tag{6}$$

where d is the size of the latent vectors $\mathbf{z}_{\mu}$ and $\mathbf{z}_{\sigma}$. z_{μ_m} and $z_{\sigma m}$ are hidden neurons at the index m in the vectors of $\mathbf{z}_{\mu}$ and $\mathbf{z}_{\sigma}$, respectively.

2) Reconstruction Loss ($\mathcal{L}_{rec}$): The reconstruction loss has a two-part form as follows:

$$\mathcal{L}_{\text{rec}} = \lambda_{\text{obj}}\left[\mathbf{x} \odot (\mathbf{x} - \hat{\mathbf{x}})^2\right] + \lambda_{\text{noobj}}\left[(\mathbf{1} - \mathbf{x}) \odot (\mathbf{x} - \hat{\mathbf{x}})^2\right], \tag{7}$$

where $\mathbf{x}$ and $\hat{\mathbf{x}}$ are the original input and the reconstructed sample vectors, respectively. $\odot$ indicates the element-wise multiplication. The motivation to separate the reconstruction into two parts is to emphasize the reconstruction of active and passive grids, separately. We have the scalar hyper-parameters of λ_{obj} and λ_{noobj} to have a weighted reconstruction loss. Note that if λ_{obj} and λ_{noobj} both equal to 1, then $\mathcal{L}_{\text{rec}}$ is equivalent to the standard mean square error (MSE).

3) Soft f1-Score Loss ($\mathcal{L}_{soft\text{-}F1}$): There are two pitfalls related to using the grid representations as input and having a sampler in the network. Firstly, a grid representation of a scene has a sparse tensor form due to the scattered layout of objects in an environment. Moreover, grid representation matrices for the objects that are not presented in a scene are empty (zero) matrices. Therefore, the number of passive cells (i.e., cells with the value of 0) outnumbers the active cells (i.e., cells with the value of 1). This fact causes an imbalance during the reconstruction. Secondly, adding independent Gaussian noise during the sampling step causes the network to produce noisy results. To overcome these problems, we introduce the soft-f1 objective term to improve a harmonic mean of precision and recall defined on the input and the reconstructed grid representations. Although the standard f1-score is not differentiable due to the decision threshold (i.e., a binary decision for being a positive or negative result), we can directly use a grid sigmoid activation as a score of positiveness and negativeness. As a result, the soft-f1 loss term is differentiable and allows back-propagation for the training.

Firstly, for a reconstructed grid representation $\hat{\mathbf{x}}$ and ground-truth input vector $\mathbf{x}$, we define soft true positive (TP), false positive (FP), false negative (FN) scores as follows:

$$\begin{aligned} TP &= \frac{1}{T}\sum_{i=0}^{r}\sum_{j=0}^{c}\sum_{k=0}^{o}\sigma(\hat{x}_{ijk})x_{ijk}, \\ FP &= \frac{1}{T}\sum_{i=0}^{r}\sum_{j=0}^{c}\sum_{k=0}^{o}\sigma(\hat{x}_{ijk})(1 - x_{ijk}), \\ FN &= \frac{1}{T}\sum_{i=0}^{r}\sum_{j=0}^{c}\sum_{k=0}^{o}(1 - \sigma(\hat{x}_{ijk}))x_{ijk}, \end{aligned} \tag{8}$$

where r, c, o represents the number of the rows, columns, and depth of an input vector $\mathbf{x}$. x_{ijk} represents the grid cell with indices i, j, k. T is a normalization constant where $T = r \times c \times o$. σ is the sigmoid function where $\sigma(x) = \frac{1}{1+e^{-x}}$ to map the value of reconstructed sample between 0 and 1. Note that the ground-truth cell x_{ijk} is either 0 or 1. Therefore, the product of two terms can be 1 at maximum.

For given the definitions of Precision $= \frac{TP}{TP+FP}$ and Recall $= \frac{TP}{TP+FN}$, we derive the differentiable soft-f1 loss as follows:

$$\begin{aligned} \mathcal{L}_{\text{soft-f1}} &= \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \\ &= \frac{TP}{TP + \frac{1}{2}(FP + FN)}, \\ &= \frac{1}{T}\sum_{i=0}^{r}\sum_{j=0}^{c}\sum_{k=0}^{o}\left[\frac{2\sigma(\hat{x}_{ijk})x_{ijk}}{x_{ijk} + \sigma(\hat{x}_{ijk})}\right]. \end{aligned} \tag{9}$$

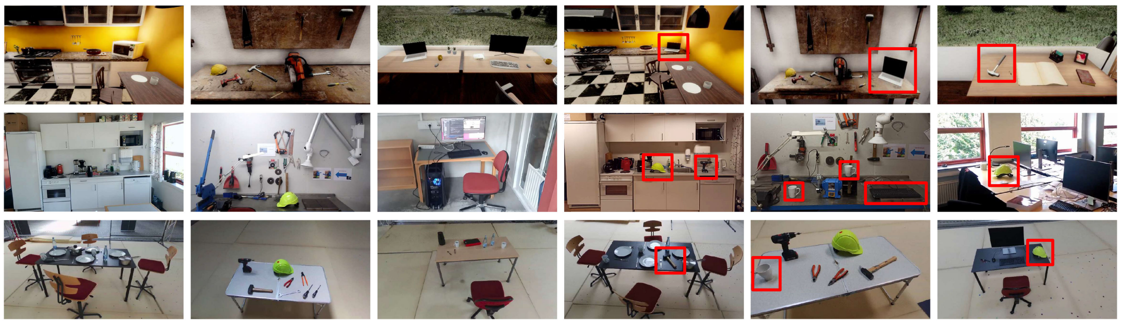


Fig. 4. Samples from our datasets. The top row shows samples from the simulated environment, the middle row shows samples for real scenes, and the bottom row shows samples from the mock-up environment. The first three columns show examples of normal (anomaly-free) samples. The last three columns are examples of test samples including anomalies. Abnormal objects are annotated with a red bounding box. [Best viewed in color].

For given 6, 7, 9, the overall loss function (grid loss) can be written as following:

$$\begin{aligned}\mathcal{L}_{\text{grid}} = \lambda_{\text{f1}} \frac{1}{T} \sum_{i=0}^{r} \sum_{j=0}^{c} \sum_{k=0}^{o} \frac{2\sigma(\hat{x}_{ijk}) x_{ijk}}{x_{ijk} + \sigma(\hat{x}_{ijk})} \\ + \lambda_{\text{kl}} \sum_{m=0}^{d} \left({z_{\sigma}}_m^{\,2} + {z_{\mu}}_m^{\,2} - \log\left({z_{\sigma}}_m^{\,2}\right) - 1 \right) \\ + \lambda_{\text{obj}} \sum_{i=0}^{r} \sum_{j=0}^{c} \sum_{k=0}^{o} x_{ijk} (x_{ijk} - \hat{x}_{ijk})^2 \\ + \lambda_{\text{noobj}} \sum_{i=0}^{r} \sum_{j=0}^{c} \sum_{k=0}^{o} (1 - x_{ijk})(x_{ijk} - \hat{x}_{ijk})^2, \quad (10)\end{aligned}$$

where λ_{f1}, λ_{kl}, λ_{obj}, and λ_{noobj} are the hyper-parameters to adjust the individual loss terms (soft f1-score loss, Kullback-Leibler loss, objectness loss, non-objectness loss, respectively) of the overall loss.

III. Experiments and Results

We extensively evaluate the presented anomaly detection approach in a comprehensive collection of simulated and real data. Firstly, we test our method in a controlled simulation environment to better understand the model's behavior. Secondly, we test our method in a challenging real environment that has scenes with different contexts and scattered object layouts. Lastly, we create a mock-up real environment to show a use-case that a drone is deployed for data collection.

In our experiments, we compare our method with the baseline methods heavily used in the literature. We choose the baselines from different approaches including an autoencoder [22], deep one-class SVM [8], GAN [23], and UAV-ADNet [2]. For the sake of compatibility, we modify the input layers of the baseline methods to take the input grids as input.

A. Dataset

We collect both simulated and real-world datasets to train and test our model and the baselines. A data sample consists of a scene image, bounding box annotations of objects that are available in the scene, and a location label for a scene. Training samples are considered normal (i.e., anomaly-free), and test samples include out-of-context objects as anomalies (See Fig. 4). We collect three datasets in total, including (i) simulated data, (ii) real data, and (iii) real mock-up data. The location labels for the scenes are obtained in different ways in each type of dataset.

Simulated Data: We use AirSim [24] as the simulation engine to create the simulation dataset. We create an environment including five rooms with different contexts such as kitchen, office, work-space, storage room (See Fig. 4). To collect training samples, we fly manually with a simulated drone to record scene images. The simulation API records scene images, ground-truth object annotations, and location vectors during the flight. The location vectors hold the x and y coordinates of the drone according to the simulated world frame. The simulation dataset includes 5140 normal samples for training and 700 anomalous samples (i.e., scenes including out-of-context objects) for testing. The number of object categories is 33 for the simulation dataset.

Real Data: In order to test our approach in a challenging anomaly detection task, we collect training and test samples from real scenes (See Fig. 4). The samples are collected from different rooms of DeepTech Experimental Hub of Aarhus University. The location vectors hold the x and y coordinates of the rooms according to the building map. The real dataset includes 12 000 training samples and 3000 test samples from 6 different rooms. The number of object categories is 70 for the real dataset.

Knowing the building plan and using a hand-held camera, we record data samples, including scene samples and room location labels, according to the global map (i.e., building map). We use two Faster-RCNN [25] object detectors that are already trained on the COCO [26] and ALET [27] datasets separately to get object annotations.

Real Mock-up Data: In the context of anomaly detection, a drone can be employed as a data source thanks to its mobility. To show a sample use case, we create a mock-up environment inside the flight cage in the Aarhus University Artificial Intelligence in Robotics Lab for the sake of safety. We create four artificial rooms belonging to specific contextual categories, including office, kitchen, and workspace (See Fig. 4). The location vectors hold the x and y coordinates of the drone according to the world frame defined by an external motion capture system. Firstly, Parrot Bebop2 is used to collect normal data for the scenes. Then, we add out-of-context objects to each scene to collect test samples. The mock-up dataset includes 4000 training samples and 1000 test samples. The number of object categories is 17 for the mock-up dataset.

B. Baselines

For comparison, we choose four baseline methods from different approaches, which are commonly used in anomaly detection tasks: (i) autoencoder, (ii) deep SVDD, (iii) GAN, (iv) UAVAdNet. These methods follow the paradigm of "learning normality" for anomaly detection and are trained on normal data in an unsupervised manner. They compare the original input and the model's reconstruction to find anomalies for the given input data.

1) Standard Autoencoder: Autoencoders are neural networks that consist of encoder and decoder parts similar to our method. In the context of anomaly detection, when an autoencoder is trained with only normal samples, they learn the encoding of normality [22]. For inference, the difference between original and reconstructed samples indicates anomalies. In our experiments, we implement the autoencoder with the same architecture with GridNet but without the residual connection for convolutional layers, and the sampling part.

2) Deep SVDD [8]: Deep SVDD is a neural network used to learn a hyperspace for a given training data. When the SVDD is trained with only normal samples, a latent representation of a normal sample fits the hyperspace, yet a latent representation of anomalous samples lies outside of the hyperspace [8]. We implement deep SVDD with the same architecture of the autoencoder.

3) Gan: GAN consists of two sub-networks, including discriminator and generator [13]. Two sub-networks are trained jointly in an adversarial manner. During training, a generator learns to produce artificial samples that are similar to the training samples, and a discriminator learns to differentiate original samples and generated samples. In our experiments, we implement the generator as an autoencoder with the same architecture in the first baseline. The discriminator is 4-layered CNN with two convolutional layers, where each layer has 16 filters with a size of 3×3 and the stride of 1×1. The convolutional layers are followed by two fully connected layers, which has 32 and 1 neurons, respectively.

4) UAV-ADnet: UAV-ADNet is a variational autoencoder-based method to find anomalies in a grid representation of an aerial image [2]. As our proposed method, UAV-ADNet takes a grid representation and GPS-location of the drone as inputs and tries to reconstruct grid representation. Contrary to the original study [2], we extract grid representations from indoor scenes instead of aerial images and use dataset-dependent location data (e.g., coordinates from the simulation engine for the simulated dataset) instead of GPS data. We follow the implementation in the original work [2].

C. Experimental Settings

We evaluate our method in simulated, real, and mock-up data, separately. Before the training and the testing, we extract grid representations for given scene images and object annotations. To this end, we choose the width and the height of a grid representation (i.e., the number of cells in a row and a column, respectively) as 32 and 24 for the simulation and mock-up samples to keep the original aspect ratio of the scene images (4:3). To keep the aspect ration of real samples (16:9), we choose the width and the height of a grid representation as 32 and 18, respectively.

For the training of all models, we use Adam optimizer with the default parameters (i.e., the initial learning rate is set 0.001; beta1 and beta2 are set to 0.9 and 0.999, respectively). The batch size is set to 32, and the training is stopped when the validation reconstruction error starts to increase. We keep the training settings the same for the experiments in simulation, real, and mock-up data.

We train GridNet to optimize the proposed grid loss. We empirically set λ_{f1}, λ_{kl}, λ_{obj}, and λ_{noobj} to 2, 1, 2, 1, respectively. GAN, Deep SVDD, and UAV-AdNet are trained to optimize the loss functions used in their original works. We use MSE as the loss function for the autoencoder. For all models, we use the same loss functions during the training on three datasets.

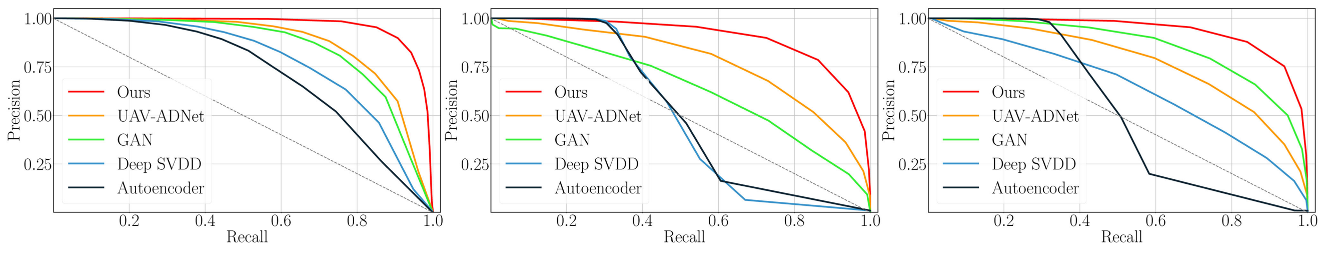
The output of GridNet (i.e., the reconstructed sample) is a tensor of real numbers. A threshold should be applied to the model's output to decide which objects are present in the reconstruction. However, thresholding causes a trade-off between precision and recall, which is determined by the threshold that separates the active grid cells from the negative grid cells. Therefore, we compute precision-recall pairs for different thresholds and plot the precision-recall curves to see the model's reconstruction performance on the test sets for comparison (See Fig. 5). The models are expected to remove anomalous objects from the reconstructed data (i.e., setting the grid value to zero). In contrast, they should recover the positive grid cells indicating the normal objects in an environment.

D. Results

Fig. 5(a) shows the precision-recall curve for the simulated data. In comparison to the other models, we see that GridNet provides the best performance. As shown in Fig. 5(a), the trade-off between the recall and the precision becomes problematic when the recall goes above 0.8 for GridNet. However, the trade-off starts at earlier points for other baselines. This shows that GridNet has more robust reconstruction ability for the anomaly detection task.

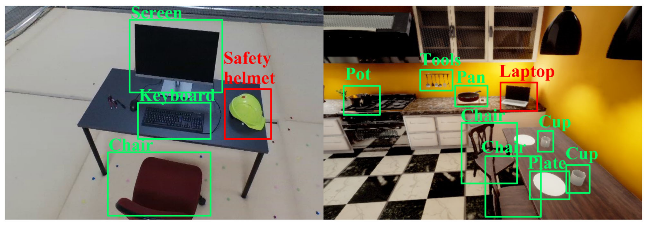
Fig. 5(b) shows the precision-recall curve for the real-world test set. As can be seen in Fig. 5(b), GridNet is significantly better than other baselines. However, the reconstruction performances of all models drop significantly for the anomaly detection task in the real dataset. This is expected since the real samples are more challenging than the simulation and mock-up samples since they include high numbers of objects with scattered layouts. Therefore, the networks should recall a higher amount of normal samples compared to simulation and real data.

From Fig. 5(c), we see that GridNet performs the best for anomaly detection in mock-up test samples, as well. The results of the models on the mock-up data are better than the results in the real data. We consider that the reason is the simplicity



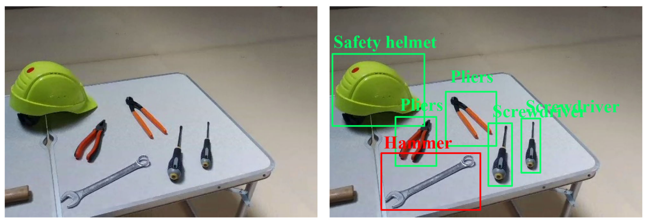
(a) Simulated data (b) Real data (c) Mock-up data

Fig. 5. Precision-recall curves for the simulated, real, and mock-up data. Note that both high precision and high recall are our desired operating domains since the network should recover the normal objects, which are already presented in the scene, and should not add any object that are not available in the scene. The black dashed diagonal line indicates the chance level.



(a) Mock-up office scene (b) Simulated kitchen scene

Fig. 6. Qualitative results from our tests. (a) and (b) illustrate the inferences of GridNet on arbitrary test samples from the mock-up and the simulation data, respectively. Red bounding boxes show the detected anomalies, whereas green bounding boxes show the normal objects.



(a) (b)

Fig. 7. GridNet can fix false predictions of an object detector for a given input image (a). The object detector mislabels the wrench as "hammer". (b) GridNet predicts the hammer as an anomaly since it is an out-of-context object according to the training scenes.

of the mock-up environment compared to the real samples. The qualitative results can be seen in Fig. 6.

As can be seen in Fig. 5, GridNet and UAV-ADNet are better than other baselines on the simulated and real-world data. The location vector allows the networks to learn the normality of a scene conditioned on its location. This is important for simulated and real-world data, where contextually similar scenes (e.g., two rooms with the office context) are present. The grid representations of two scenes are same if they include the same object types with the same layout in their scene images. In that case, the location vector enables the networks the differentiate the scenes. Other baselines cannot differentiate grid representations of two contextually similar scenes since they do not accept the location vector as input. This case is not valid for the mock-up data since the scenes are contextually distinct in this dataset.

In addition to the anomaly detection tests, we test the effect of the grid loss. For this end, we train GridNet with MSE and ELBO loss. For testing, we calculate the reconstruction error as the average of the test reconstruction errors of three datasets. The network has the lowest reconstruction loss when it is subject to minimize the grid loss (0.02) compared to MSE (0.18) and ELBO (0.09) losses.

IV. Discussion

GridNet is image-agnostic since it cannot get any image feature during training and testing phases. Therefore, it can work on top of any object detector. Although this allows our model to have the flexibility of choosing any object detector, its result can be affected by the poor performance of object detectors. In our qualitative results, we observe that our method can decrease false-positive predictions of object detectors (See Fig. 7). Therefore, GridNet can be suggested to improve object detection results eliminating false positive predictions. Note that object detection is not the focus of this letter, and we do not try to optimize object detectors.

GridNet is image-agnostic yet object detectors might still be vulnerable to vision-related problems. As our indoor scenarios do not contain significant visual challenges thanks to consistent illumination, the object detectors trained with generic datasets (e.g., COCO, ALET) are sufficient to find objects. However, the object detectors might need to be improved for more challenging environments such as outdoors.

We also note that GridNet can find anomalies among the detection results of an off-the-shelf object detector. Therefore, a novel object that is out of the object detector vocabulary cannot be detected as an anomaly. In this work, we consider anomalies as the presence of objects that should not be in a specific scene. Being limited with an object vocabulary, detecting novel objects, which are unknown for the object detector, is not our focus.

We use the reconstruction of meta representations of images to find anomalies instead of raw image reconstruction as in similar studies. Although our method allows an abstraction from vision-related issues, it is limited for finding anomalies caused by object appearance changes since grid representations contain information regarding the presence of objects yet not their appearance.

Having the grid representation might have a disadvantage when object instances with the same category are too close. In that case, the centers of two objects with the same category might fall into the same grid cell; however, a grid cell can represent only one object instance per category. Therefore, the image should be divided into a higher number of grids if the objects are too close. This case is not a problem for close objects with different categories since they are represented with different grid cells.

In our experiments, we use global coordinates available in our three experimental setups as location vectors. However, having a consistent location vector for an environment might not be straightforward and require specific settings constraints such as having a global map of an environment and an accurate localization algorithm for a drone. Moreover, accurate detection of all interested objects might require a fine-tuning of object detectors in practice.

Although we conduct our experiments in indoor environments, our approach could be extended to outdoor surveillance scenarios such as traffic monitoring, surveillance of critical infrastructures. In that case, GPS coordinates could be used as a location vector. However, drastic changes in illumination might affect object detection results.

This study presents that GridNet trained with the grid loss outperforms the selected baselines for anomaly detection. These methods are not proposed for grid representations originally, and we modify these methods to become compatible with our approach. A more detailed analysis of the hyper-parameter search for the baseline models can be conducted as future work.

V. Conclusion

In this work, we propose an unsupervised anomaly detection method (GridNet) that is suitable for indoor surveillance. Our method can learn the normality of a scene conditioned on the scene location on a global plan. Unlike similar studies, GridNet is image-agnostic and accepts grid representations of scenes instead of raw images. Therefore, it is not affected by vision-related problems. We also propose a new loss function that is suitable for anomaly detection using grid representations. We show that the new loss function improves the model's exactness and completeness. In our experiments, we compare our method with existing approaches on both simulated and real data, and show our method's superiority.

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009.
[2] I. Bozcan and E. Kayacan, "UAV-ADnet: Unsupervised anomaly detection using deep neural networks for aerial surveillance," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 1158–1164.
[3] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proc. 2nd Int. Conf. Learn. Represen.*, 2014.
[4] G. Pang, C. Shen, L. Cao, and A. v. d. Hengel, "Deep learning for anomaly detection: A review," 2020, *arXiv:2007.02500.*
[5] K. Doshi and Y. Yilmaz, "Fast unsupervised anomaly detection in traffic videos," in *Proc. IEEE/CVF Conf. on Comput. Vis. Pattern Recognit. Workshops*, 2020, pp. 624–625.
[6] V. Saligrama and Z. Chen, "Video anomaly detection based on local statistical aggregates," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2112–2119.
[7] D. Xu, E. Ricci, Y. Yan, J. Song, and N. Sebe, "Learning deep representations of appearance and motion for anomalous event detection," *Proc. of BMVC*, 2015, pp. 8.1–8.12.
[8] L. Ruff *et al.*, "Deep one-class classification," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4393–4402.
[9] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 733–742.
[10] D. Gong *et al.*, "Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1705–1714.
[11] A. Adler, M. Elad, Y. Hel-Or, and E. Rivlin, "Sparse coding with anomaly detection," *J. Signal Process. Syst.*, vol. 79, no. 2, pp. 179–188, 2015.
[12] W. Luo, W. Liu, and S. Gao, "A revisit of sparse coding based anomaly detection in stacked RNN framework," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 341–349.
[13] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "GANomaly: Semi-supervised anomaly detection via adversarial training," in *Proc. Asian Conf. Comput. Vis.*. Springer, 2018, pp. 622–637.
[14] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat, and V. Chandrasekhar, "Adversarially learned anomaly detection," in *Proc. IEEE Int. Conf. Data Mining*, 2018, pp. 727–736.
[15] I. Bozcan and S. Kalkan, "COSMO: Contextualized scene modeling with boltzmann machines," *Robot. Auton. Syst.*, vol. 113, pp. 132–148, 2019.
[16] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly detection in crowded scenes," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 1975–1981.
[17] J. Kim and K. Grauman, "Observe locally, infer globally: A space-time mrf for detecting abnormal activities with incremental updates," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 2921–2928.
[18] I. Bozcan, Y. Oymak, I. Z. Alemdar, and S. Kalkan, "What is (missing or wrong) in the scene? A hybrid deep boltzmann machine for contextualized scene modeling," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1–6.
[19] R. Hinami, T. Mei, and S. Satoh, "Joint detection and recounting of abnormal events by learning deep generic knowledge," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3619–3627.
[20] I. Bozcan and E. Kayacan, "AU-AIR: A multi-modal unmanned aerial vehicle dataset for low altitude traffic surveillance," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 8504–8510.
[21] S. Odaibo, "Tutorial: Deriving the standard variational autoencoder (VAE) loss function," 2019, *arXiv:1907.08956*.
[22] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proc. 2nd Workshop Mach. Learn. Sensory Data Anal.*, 2014, pp. 4–11.
[23] 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 *Proc. Int. Conf. Inf. Process. Med. Imag.*. Springer, 2017, pp. 146–157.
[24] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "AirSim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field Serv. Robot.*. Springer, 2018, pp. 621–635.
[25] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[26] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*. Springer, 2014, pp. 740–755.
[27] F. C. Kurnaz, B. Hocaoğlu, M. K. Yülmaz, I. Sülo, and S. Kalkan, "ALET: A dataset, a baseline and a usecase for tool detection in the wild," in *ECCV*., Springer, Cham, 2020, pp. 371–386.