Unsupervised Anomaly Detection in Sewer Images with a PCA-based Framework

Dirk Meijer
LIACS
Leiden University
Leiden, Netherlands
meijerdwj@liacs.leidenuniv.nl

Mitchell Kesteloo
LIACS
Leiden University
Leiden, Netherlands

Arno Knobbe
LIACS
Leiden University
Leiden, Netherlands

***Abstract*—We propose a simple framework for unsupervised anomaly detection in aligned image sets, consisting of (i) feature extraction, (ii) PCA decomposition and partial reconstruction, and (iii) a dissimilarity measure comparing reconstructed to extracted features. The basic principle is explained using artificial datasets, and we show the effectiveness in a real-world scenario pertaining to sewer inspection images. We investigate the effectiveness of several features for this specific task, and show that concatenating several features results in superior performance. An analogy is also drawn to convolutional autoencoders and we compare to some simplistic renditions of such networks to our framework.**

Anomaly detection, Principal component analysis, Convolutional autoencoder, Unsupervised learning, Image processing, Sewer asset management

I. Introduction

Sewer inspections need to be performed periodically to ensure performance is up to standards. Image and video data collected by a 'pipe inspection gadget' (PIG) is often manually inspected, leading to subjective, inconsistent, and unreliable deterioration and urgency ratings [1]. The SewerSense project [2] aims to automate such sewer inspections.

Anomaly detection (sometimes outlier or novelty detection) is a problem which aims to detect observations that do not conform to an expected pattern. In the context of image and video data, this relates to finding regions of interest (ROIs), portions of the video or images that have a different appearance and might be of interest. In the context of sewer inspections, automatic ROI detection is the first step of the SewerSense project. At a later stage in the project, classification of the found regions into a taxonomy of defect classes will be considered, but the intermediate result should aid the inspection process in itself.

In this work, we propose a three-part framework to detect anomalies in *aligned image sets*, such as static camera video or photographs, or registered images. The framework is based on principal component decomposition and partial reconstruction, but accounts for the fact that not all common elements in image sets can be accounted for by a linear model (such as PCA is) by first extracting possibly non-linear features from the image sets. We also foray into the field of deep learning and investigate the possibility of using convolutional autoencoders (CAEs) to fill the role of several parts of the framework.

We would like to emphasize that while this work originated from the need to automatically process sewer pipe images, no assumptions are made specific to this problem. The only requirement is that *the images in a set are aligned*, so other possible applications include video surveillance, autonomous vehicles and medical image processing.

This work is part of the Cooperation Programme TISCA (Technology Innovation for Sewer Condition Assessment) with project number 15343, which is (partly) financed by NWO domain TTW (the domain applied and Engineering Sciences of the Netherlands Organisation for Scientific Research), the RIONED Foundation, STOWA (Foundation for Applied Water Research) and the Knowledge Program Urban Drainage (KPUD).

II. Preliminaries

A. Principal Component Analysis

Principal Component Analysis (PCA) has been around for over a century, after having been introduced in 1901 by Karl Pearson [3]. It is a popular tool in statistics, data science and many other scientific fields, used to reduce the dimensionality of data to facilitate data exploration and the use of algorithms that are sensitive to high dimensionality.

Given a dataset $\mathbf{X}$, consisting of N observations with d features each, we express this as an $[N \times d]$ matrix. When using PCA for dimensionality reduction, a dimensionality $\theta \leq d$ is chosen and $\mathbf{X}$ is projected on the first θ eigenvectors The projected matrix $\mathbf{P}$ retains as much variance as is possible in θ dimensions*. This allows researchers to view high-dimensional data in two or three-dimensional plots, or employ algorithms that are not designed for high-dimensional data.

B. Anomaly Detection

In this study, we approach anomaly detection as an unsupervised learning problem. This means the algorithm will learn the structure present in the data, with the caveat that some parts of the data will not adhere to this structure. These anomalies have to be detected without influencing the

*Barring non-linear embeddings

Algorithm 1: Anomaly Detection Framework

Input : Image set $\{\mathbf{I}_1, \ldots, \mathbf{I}_N\}$

Input : Feature descriptors $\mathbf{F} : \mathbf{I} \rightarrow \mathbb{R}^d$

Input : Number of principal components to use in reconstruction: θ

Input : Dissimilarity function $\mathbf{D} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_+$

Initialize: Featurespace $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ with $\mathbf{x}_i \in \mathbb{R}^d \quad \forall i \in [1, N]$

Initialize: Featurespace $\mathbf{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$ with $\mathbf{p}_i \in \mathbb{R}^d \quad \forall i \in [1, N]$

Initialize: Featurespace $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N\}$ with $\hat{\mathbf{x}}_i \in \mathbb{R}^d \quad \forall i \in [1, N]$

1 $\mathbf{x}_i \leftarrow \mathbf{F}(\mathbf{I}_i) \quad \forall i \in [1, N]$ `// Extract per-image features`

2 $\mathbf{P} \leftarrow \text{PCA}(\mathbf{X})$ `// Decompose and project X onto its PCs`

3 $[\mathbf{p}_i]_j \leftarrow 0 \quad \forall i \in [1, N] \quad \forall j \in (\theta, d]$ `// Discard contribution of low variance PCs`

4 $\hat{\mathbf{X}} \leftarrow \text{PCA}^{-1}(\mathbf{P})$ `// Reconstruct to original feature space`

5 $A_i \leftarrow \mathbf{D}(\mathbf{x}_i, \hat{\mathbf{x}}_i) \quad \forall i \in [1, N]$ `// Calculate per-image anomaly score`

Output : Anomaly scores $\{A_1, \ldots, A_n\}$

learning of the structure too much. This draws some parallels to one-class classification and clustering. The difference in one-class classification is the assumption that the dataset used to learn the structure will be pure, that is to say, it does not contain the anomalies that are to be detected after learning [4]. The difference with clustering is that clustering cares about different classes of objects that may be present in the dataset, while anomaly detection only looks for the anomalies, regarding all non-anomalous objects as a single background class. Anomalies might also be single instances, meaning they would not be considered as clusters.

Anomaly detection is a large field, but an especially well-written and extensive overview of methods, empirical issues, and literature is given in [5]. While using PCA to detect anomalies is not a new idea (see [6] for example), to the authors' best knowledge the application to image data is novel.

III. Framework

We propose a simple three-part framework to detect local anomalies in aligned image sets and videos, as shown in figure 1 and described in more detail in algorithm 1. The three parts are: (i) feature descriptors, (ii) PCA decomposition and partial reconstruction, (iii) a dissimilarity function to compare the PCA reconstructed feature to the extracted features.

a) Feature Descriptors: The choice of feature descriptor depends on the type of anomaly that has to be detected in the images. For example, to detect abnormal texture, we might use a feature that is known to work well in texture classification such as wavelet responses [7]. Or to detect motion in otherwise static camera images, we might calculate the difference between a frame and the previous frame at each position. The simplest choice is an identity function, i.e. the features are the original pixels in the image.

The reason for using feature descriptors instead of simply the images themselves stems from the fact that PCA is a linear model, and the resulting principal components will be combined linearly to reconstruct each image. Extracted features, unlike the images they came from, may have invariances to transformations that makes them more suited to compare images within a certain set than the raw pixel values would.

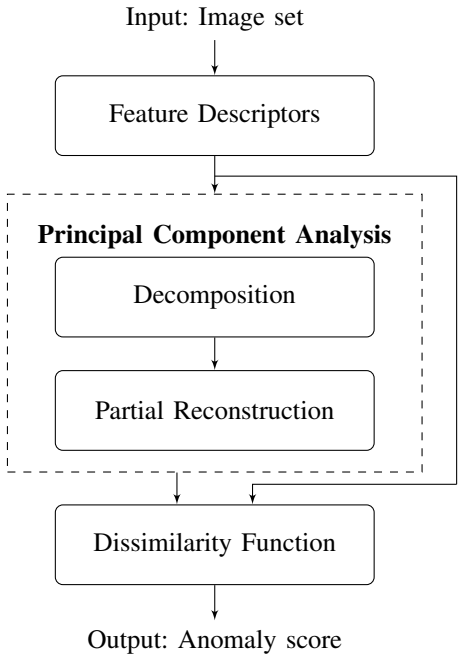

Fig. 1. The proposed three-part anomaly detection framework.

A feature can be used to describe an entire image, a specific location, or portions of an image, depending on the descriptor used. This determines how 'localized' the anomaly detection is. For example, we might calculate a locally windowed greyscale histogram, resulting in as many feature vectors as we have windows for each image in the set. We might want to detect entire images as being anomalous, or we might want to focus on specific regions within the image. When using localized features, we have the option to either treat all resulting feature vectors as if they came from the different

images (treating each window location as an image in itself) or perform the framework for each window location individually.

b) PCA Decomposition and Partial Reconstruction: The core of this approach is PCA decomposition and partial reconstruction. The rationale is as follows: Common structure within the image set will account for a large amount of the variance present in the set. By decomposing the feature vectors into principal components and discarding components that represent less occurring variations before performing partial reconstruction, we are using PCA akin to a *trained image smoother*, which keeps common and discards uncommon structure.

This step requires a parameter θ, the number of principal components used for reconstruction. This parameter corresponds roughly to a bias/variance trade-off. A very high θ might result in overfitting on the training set as the difference between original and reconstructed feature vectors might contain mostly noise, and processing an image that the method was not trained on would not work well. A very low θ means the method relies more on deviations from the mean feature vector, and less on the deviations it might learn from the training set. It is also possible to replace this abstract parameter θ with a more interpretable concept by choosing a percentage of explained variance that the model should learn, and setting θ to the lowest number of principal components that explain at least that amount of variance.

c) Dissimilarity Function: To determine whether something is or isn't an outlier, the decomposed and reconstructed feature vector is compared to the extracted local feature vector by means of some dissimilarity function. This might be Euclidean distance, (one minus) a normalized Pearson correlation, or however the chosen feature descriptors are usually matched in other applications. It should be noted that PCA minimizes the mean squared reconstruction error, so this is also minimized for the anomalies we want to detect. It can be any function $D(f_1, f_2)$ that compares two feature vectors f_1 and f_2, with only the restrictions that $D(f_1, f_1) = 0$, the dissimilarity of a feature vector to itself is zero, and $D \geq 0$ for all feature vectors.

We call the dissimilarity of the feature vector to its partial reconstruction the *anomaly score*. This anomaly score can then be thresholded to determine whether each feature vector represents an anomalous image or region.

A. Proof of Concept

To illustrate our method, we look at the MNIST reference dataset [8], consisting of 70,000 handwritten digits in greyscale images of dimensions $[28 \times 28]$ We use the identity function as feature descriptor, so that the feature vector is identical to the pixel vector. This means our feature matrices are shaped $[70000 \times 784]$. When we apply PCA to the MNIST dataset, we obtain 784 principal components, which we can reshape into $[28 \times 28]$ images for visual inspection (also known as *eigenimages*), as shown in figure 2 for the first 9 principal components.

Fig. 2. The mean values (left) and first 9 principal components of the MNIST dataset. (Greyscale ranges have been rescaled for maximum visibility.)

Now when we project an image onto the basis spanned by the principal components, we express the image as a linear combination of the eigenimages. Since the eigenimages are sorted in order of decreasing explained variance in, an image that is similar to the images in the set (in this case: also a handwritten digit, for example) is expected to have a larger (absolute) projected component onto earlier principal components (higher eigenvalues), than onto later principal components (lower eigenvalues).

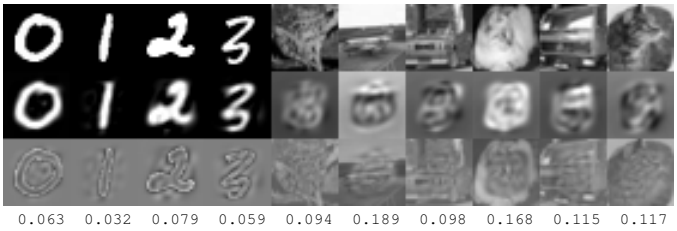

Fig. 3. Sample images from the MNIST and CIFAR-10 datasets (top row) are reconstructed with the first 50 principal components after PCA was trained on 70,000 MNIST images and 1,000 CIFAR-10 images (middle row) and the difference images between the original and the reconstructions (bottom row). Below each difference image is the mean absolute value, which is used as the anomaly score.

Now what happens when our dataset contains anomalies? To illustrate, we add the first 1,000 images of the CIFAR-10 dataset of natural images [9] to the MNIST dataset[†]. These images are very different from the digits in the MNIST set, and since there are so few of them compared to the total size of the dataset, they can be considered anomalies. We train PCA on the combined dataset and then recreate all images using only the first 50 principal components. We show the reconstruction of some sample images in figure 3. It can be seen here that the images from the CIFAR-10 set reconstruct poorly at the edges, which makes sense as 98.5% of the images are from the MNIST dataset, which does not contain any structure on the edges of the images. As a result, the difference images contain more structure at the edges and the CIFAR-10 images will be easier to distinguish from the MNIST images with our dissimilarity function.

As dissimilarity function, we take the mean absolute value of the pixels in the difference images, which gives us an anomaly score for each image in the set, which is on average going to be higher for images from the CIFAR-10 dataset than images from the MNIST dataset (see for example the anomaly scores of the example images in figure 3). We can now predict which images are anomalies by thresholding the anomaly score.

[†]The images from CIFAR-10 are converted to greyscale and cropped to $[28 \times 28]$ pixels to conform to the images in the MNIST set.

This illustrates the basic principle of the framework: the reconstruction error with a limited number of principal components can find anomalies in an image set of otherwise similar appearance. Although no feature descriptors were used for this simple example, the need for this will become clear in the next chapter.

IV. Application in Sewer Pipe Images

Dutch sewer engineering company vandervalk+degroot has provided us with a dataset of images from a front-facing camera on a PIG (pipe inspection gadget), from ten different streets within different municipalities in the Netherlands. These images are already spatially aligned, as the inspector has aligned the camera to the center of the pipe before starting. The images contain no labels or annotations though, so a method of verifying that the unsupervised method correctly finds anomalies is required. To this end, we selected two different subsets that are somewhat representative of all the sewer pipes from the different municipalities present in the datasets and hand-labeled 22 images from these sets.

The two subsets correspond to two different types of pipe: (1) smooth concrete and (2) more rough and textured agglomerate. Figure 4 shows an example of both. Henceforth, we will refer to these two image sets as 'smooth' and 'coarse'. The image sets contain 684 and 698 images respectively.

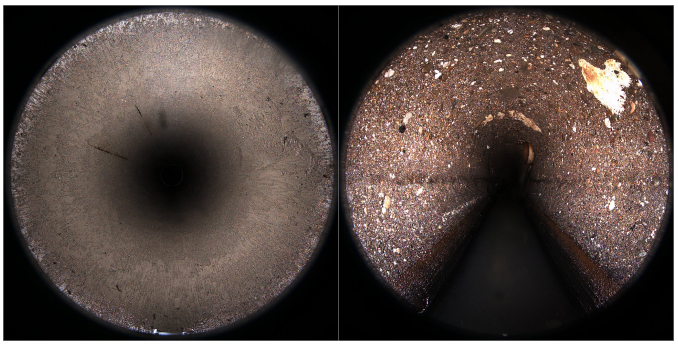

Fig. 4. Sample images from the two labeled datasets: on the left the more smooth concrete pipe, on the right the more roughly textured agglomerate.

The images are processed by the framework on a per-street basis. The reason for this is that the material used varies for different municipalities and date of installation, as will the effects of age. When using images from a single street, we can be reasonably certain that all images in such a set are of similar appearance, which means that anomalies are more easily detected, because we do not have to account for a possible multimodal distribution in appearance.

The images were divided into 324 patches of $[40 \times 40]$ pixels in the regions of the image that are in the focused portion of the images. Each patch in the 22 validation images was labeled as 'anomaly' or 'normal', in the context of the rest of the pipe. This includes both actual defects, such as discoloration as a result of leakage, as well as physical features that are simply less common than others, such as pipe joints and refuse.

All the images in the sets from which the labeled images originate are divided into the same 324 patches as the labeled images, and for each patch location features are extracted and PCA is applied to the feature vectors at a specific location. This means the framework is applied 324 times and each patch location across the images is treated as a separate image set. We construct an ROC curve by thresholding the anomaly scores at various levels and obtaining true and false positive rates for our labeled validation images. We report the area under the ROC curve (AUROC) as a measure of how well the resulting anomaly score performs.

Note that the parameter θ, the cutoff value for the number of principal components to use in reconstruction, was chosen to maximize the AUROC. In our experiments, we found that the optimal value for θ corresponds to approximately 99% explained variance for the smooth image set and 95% explained variance for the coarse image set.

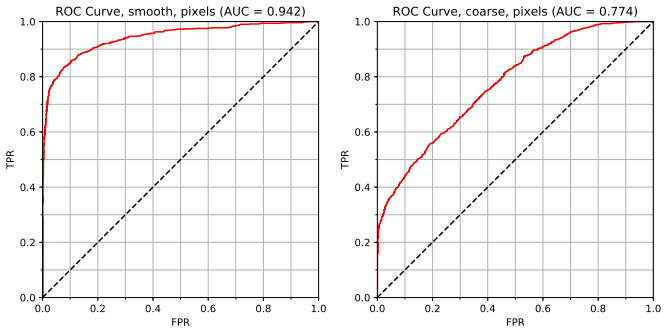

Fig. 5. ROC curves we obtain from the anomaly detection framework on our manually labeled validation set, using pixels as features to be analyzed by PCA. On the left the smooth dataset, on the right the coarse dataset.

When using pixels as features and the mean absolute difference as a dissimilarity measure, we obtain results as shown in figure 5. The AUROC for the smooth set is 0.942, high enough to be of use, however, the AUROC for the coarse set, 0.774, is quite low. This means that if this method were to see use, human operators would have to sift through a large amount of false positives if the system were calibrated to an acceptable true positive rate.

The reason the framework performs less well on the coarse set when using pixels as features, is the texture present in the surface of the pipe in those images. The variance between pixel values is far greater than it is in the smooth set, where the entire pipe is more or less a single color, and as a result the image are difficult to capture in a linear model such as PCA.

A. Feature Extraction

To alleviate this issue, we extract features that we hope are more robust to textured images. The feature vectors are then decomposed, reconstructed and compared in the same way that the images would be, as shown in the framework in figure 1. In this section, we propose five higher-level features. An overview of each features' invariances is given in table I.

TABLE I
OVERVIEW OF FEATURE EXTRACTORS INVARIANCES AND TYPICAL USES

Feature	Invariances
Pixel Values	None
Color Histogram	Translation, rotation, scaling
Fourier Transform	Translation‡
Histogram of Oriented Gradients	None
Local Binary Patterns	Translation, rotation
Homogeneous Texture Descriptor	Translation, rotation

1) Color Histograms: To illustrate the effectiveness of feature extraction, we choose a simplistic feature: a histogram of the pixel values. The 1600 values in each color channel of a patch are binned into 20 equally sized bins and concatenated to form a feature vector of length 60. These (in comparison) small vectors are decomposed into principal components and reconstructed with fewer than 60 principal components. The histogram is compared to the reconstructed histogram again by mean absolute difference. We see a slight improvement when using the histograms on the coarse set, an AUROC of 0.790, whereas performance on the smooth set is slightly worse with an AUROC of 0.942.

2) Fourier Transform: We perform a 2-dimensional Fourier transform on the $[40 \times 40]$ image patches, obtaining the frequency representation of the image patches. We discard the phase component by taking the absolute value and discard half the frequency plane because of symmetry. Again we decompose and try to reconstruct the feature vector, using the mean absolute difference as dissimilarity measure. The Fourier transform does not provide an improvement over using the pixel values, as we obtain an AUROC of 0.928 on the smooth set and 0.715 on the coarse set.

3) Histogram of Oriented Gradients: Often abbreviated as HOG, histograms of oriented gradients [10] describe an image by determining gradient directions at each pixel location, and binning these locally into histograms over a patch of specified size. It is often used for object recognition. It seems that this feature does not suit our purpose too well, as the AUROC for the smooth set becomes 0.886 and for the coarse set becomes 0.588. This can be explained by the fact that this feature is meant for object detection, and our images contain mostly texture.

4) Local Binary Patterns: Local binary patterns are a feature used to describe points as being edges or corners [11]. Each pixel is compared to its neighboring n pixels (usually $n = 8$) and for each of these neighbors, it assign a 1 or 0 depending on whether the pixel has a higher value than that particular neighbor. The resulting 8-bit numbers are locally binned to summarize the texture of a cell as containing corners, edges, or otherwise. The concatenated histograms are used as a feature vector. We obtain AUROCs of 0.865 for the smooth set and 0.705 for the coarse set.

‡After discarding phase component

5) Homogeneous Texture Descriptor: Part of the MPEG-7 multimedia description standard, homogeneous texture descriptors are shown to perform well on image retrieval tasks, even for images with much texture [12]. Simply put, the HTD features are comprised of (logarithmically scaled) mean values and standard deviations of Gabor wavelet responses.

B. Concatenating Feature Vectors

One of the strengths of the framework is that we can concatenate multiple feature vectors and the PCA reconstruction will still function identically. This allows us to combine the strengths of multiple feature types, and even combine these with the raw pixel values if we wish to do so.

After examining every possible permutation of the features previously described, we found that excluding the HOG and Fourier transform from the feature vector gave the best result on both image sets. Figure 6 shows the resulting ROC curves when we use the other high-level features described in this section, as well as the raw pixel values, giving us the highest AUROCs so far, 0.950 for the smooth set and 0.818 for the coarse set.

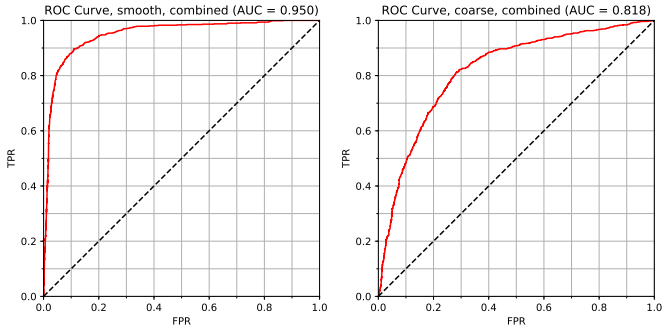

Fig. 6. ROC curves from the anomaly detection framework on the validation set, using both pixel values and the high-level features described in this section (except for HOG and Fourier transform) combined as features to be analyzed by PCA.

V. CONVOLUTIONAL AUTOENCODER

An autoencoder is a neural network that tries to learn the identity function [13], and a convolutional autoencoder combines this with image filter learning. Analogous to our framework, this means we can learn the feature representation, perform non-linear dimensionality reduction (replacing the PCA) and reconstruct the input images. As we train this network on an image set, we should be similarly able to use it to detect anomalous regions by inspecting the difference image.

We designed a convolutional autoencoder consisting of:

- Input layer: $[1040 \times 1040]$ resolution
- Convolutional layer 1: 10 $[20 \times 20]$ filters, stride $[10 \times 10]$
- Pooling layer 1: $[2 \times 2]$ max pooling, stride $[2 \times 2]$
- Convolutional layer 2: 10 $[20 \times 20]$ filters, stride $[10 \times 10]$
- Pooling layer 2: $[2 \times 2]$ max pooling, stride $[2 \times 2]$
- Autoencoder: $1690 \rightarrow 845 \rightarrow 422 \rightarrow 845 \rightarrow 1690$ units
- Unpooling layer 1: uniform, $[2 \times 2]$
- Deconvolutional layer 1: Weights shared Conv. layer 2

- Unpooling layer 2: uniform, $[2 \times 2]$
- Deconvolutional layer 2: Weights shared Conv. layer 1
- Output layer: $[1040 \times 1040]$ resolution

Using this network, trained on the same image sets, we obtained the following results: an AUROC of 0.946 on the smooth set and 0.714 on the coarse set, figure 7 shows the ROC curves. The results on the smooth set are rather similar to those obtained by the PCA framework, the AUROC results on the coarse set are noticeably worse.

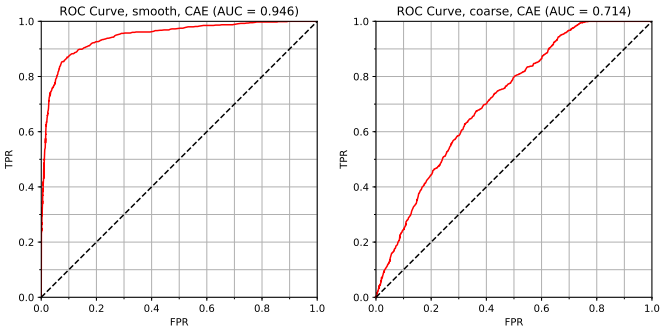

Fig. 7. ROC curves from convolutional autoencoder.

One niche where the convolutional autoencoder would outperform the PCA-based method is when we cannot afford to miss any potential defects: we can see from comparing the ROC curves that the convolutional autoencoder reaches a true positive rate of 1.0 at a lower false positive rate than the PCA-based method. Overall performance is still expected to be worse, as indicated by the AUROC.

We expect that the reason for this reduced performance is the reconstruction of the full images. In the PCA framework, we are extracting features, decomposing and reconstruction these features, and comparing the reconstruction to the *extracted features*. In the convolutional autoencoder, we try to reconstruct the image itself out of necessity, as we do not know what the features should be. But this means that the reconstructed images are compared to the original images, instead of the reconstructed features to the original features.

The fact that the convolutional autoencoder has to reconstruct the original image, means it can't learn features we might describe as 'texture descriptors,' as these are inherently rotation and translation independent, so reconstructing the original pixel values from such features would be impossible. But these are the types of features we expect (and confirmed for the PCA-based approach) to perform well, so the comparison is not entirely fair.

It should also be noted that the metaparameters of the network are far more difficult to optimize than the parameter θ our framework relies on, and a network better designed for this specific task may perform better.

VI. Summary

We have proposed a framework for unsupervised anomaly detection in aligned image sets. Table II summarizes the results obtained by the different variants. We see that while raw pixel values perform quite well on the 'smooth' dataset, improvement can be made by combining different feature descriptors. For the 'coarse' dataset, the difference is larger, drastic improvements are made by combining features.

TABLE II
Results for the methods and datasets described in this work.

	AUROC	
Feature type	smooth	coarse
Pixels	0.942	0.774
Color Histogram	0.942	0.790
Fourier Transform	0.928	0.715
Histogram of Oriented Gradients	0.886	0.588
Local Binary Patterns	0.865	0.705
Homogeneous Texture Descriptor	0.941	0.785
Pixels + Histogram + LBP + HTD	**0.950**	**0.818**
Convolutional Autoencoder	0.946	0.714

We conclude that our PCA-based approach, which could be considered a more 'traditional' statistical approach to computer vision using combinations of hand-crafted features, outperforms the more 'modern' convolutional autoencoder, but we must also admit that the comparison is not entirely fair as we are in one case reconstructing high-level features and in the other case pixel values.

References

[1] J. Dirksen, F. Clemens *et al.*, "The consistency of visual sewer inspection data," *Structure and Infrastructure Engineering*, vol. 9, no. 3, pp. 214–228, 2013.

[2] TISCA programme funded by NWO-TTW, "Sewersense – multi-sensor condition assessment for sewer asset management," 2016-2020.

[3] K. Pearson, "LIII. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[4] M. David, "Tax. one-class classification; concept-learning in the absence of counter-examples," *ASCI dissertation series*, vol. 65, 2001.

[5] G. O. Campos, A. Zimek *et al.*, "On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 891–927, 2016.

[6] M.-L. Shyu, S.-C. Chen *et al.*, "A novel anomaly detection scheme based on principal component classifier," MIAMI UNIV CORAL GABLES FL DEPT OF ELECTRICAL AND COMPUTER ENGINEERING, Tech. Rep., 2003.

[7] M. Unser, "Texture classification and segmentation using wavelet frames," *IEEE Transactions on image processing*, vol. 4, no. 11, pp. 1549–1560, 1995.

[8] Y. LeCun, C. Cortes, and C. J. Burges, "The MNIST database of handwritten digits," 1998.

[9] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009, https://www.cs.toronto.edu/~kriz/cifar.html.

[10] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

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

[12] Y. M. Ro, M. Kim *et al.*, "MPEG-7 homogeneous texture descriptor," *ETRI journal*, vol. 23, no. 2, pp. 41–51, 2001.

[13] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural networks*, vol. 2, no. 1, pp. 53–58, 1989.