Encoding Semantic Attributes - Towards Explainable AI in Industry

Sarah Schneider
Human Robot Interaction Lab, Tufts University
Boston, Massachusetts , USA
High-Performance Vision Systems
Vienna, Austria
sarah.schneider@ait.ac.com

Doris Antensteiner
High-Performance Vision Systems
Vienna, Austria
doris.antensteiner@ait.ac.at

Daniel Soukup
High-Performance Vision Systems
Vienna, Austria
daniel.soukup@ait.ac.at

Matthias Scheutz
Human Robot Interaction Lab, Tufts University
Boston, Massachusetts , USA
matthias.scheutz@tufts.edu

ABSTRACT

The transformation of industrial environments is progressing at a fast pace as more and more autonomous systems are installed and operated. Save and explainable AI algorithms are thus essential, especially for collaborative interactive systems that operate in human spaces. We propose the "Semantic Encoder", a 2D-vision based CNN model trained on a purely synthetic dataset, to address the explainability aspect by extracting semantic descriptions of real objects based on their visual appearances. We can use the extracted semantic information to simply describe depicted samples or to differentiate between normal and anomalous samples, with the possibility to explain what caused the anomaly detection. The semantic description can be further used to sort samples by classifying them or to find a sample with specific semantic properties. We evaluate the Semantic Encoder with respect to its informative power by comparing the computed semantic features with features extracted by a VGG-16 model [11] and classical image processing methods. The results are quantified based on the Generalized Discriminative Value (GDV) [10]. We also investigate how accurately anomalous samples are detected by computing ROC and PR curves. We use the semantic parameters to understand what causes good and inaccurate anomaly detection decisions. In addition, we evaluate the quality of the classification based sorting by examining confusion matrices and classification accuracy.

CCS CONCEPTS

• **Applied computing** → **Computer-aided design**.

KEYWORDS

computer vision, neural networks, anomaly detection, novelty detection, machine learning

PETRA '23, July 05–07, 2023, Corfu, Greece

ACM ISBN 979-8-4007-0069-9/23/07.
https://doi.org/10.1145/3594806.3596531

ACM Reference Format:
Sarah Schneider, Doris Antensteiner, Daniel Soukup, and Matthias Scheutz. 2023. Encoding Semantic Attributes - Towards Explainable AI in Industry . In *Proceedings of the 16th International Conference on PErvasive Technologies Related to Assistive Environments (PETRA '23), July 05–07, 2023, Corfu, Greece.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3594806.3596531

1 INTRODUCTION

Autonomous AI-based systems are increasingly utilized in industrial settings to increase efficiency and performance of production and other industrial processes. However, even controlled industrial environments present challenges such as unexpected faults in the production process or infelicities with unknown causes that are not obvious and cannot be anticipated. While a standard approach

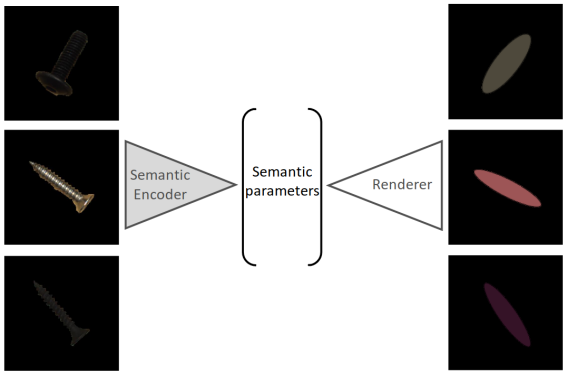


Figure 1: Illustration of some semantic descriptions of objects from the MVTech Screws [13] dataset. The rendered extracted semantic information mimics an envelope of the depicted object. The extracted colors are meaningful in the sense that for darker screws darker colors are captured, for more "reddish"-colored screws, more "reddish" colors are captured.

would be to use simulations to generate training data for AI algorithms, this approach fails in cases of unknown causes and effects.

When a defective sample is detected, the goal is to not only to report it, but to also isolate whatever caused the defect. For this purpose, being able to describe the defect or deviation from the expected outcome is critical.

Compiling training sets from industrial samples, however, comes with its own set of challenges. For example, experts are needed for accurate sample annotations. Samples without defects appear with much greater extent than anomalous samples, leading to highly unbalanced training sets for supervised approaches. Almost all machine-learning based systems make the assumption of a "closed world", assuming a complete system model[2]. However, the system might be exposed to a constantly evolving environment and needs to adapt to new settings in a trustworthy manner. Hence, an "open-world" approach to defect detection is needed that allows for characterizing unexpected and unknown visual features in a way that human operators can understand and that allows them to trace the effects back to their causes in the fabrication process. The goal of this paper is to present such a system and show its promise through a sequence of evaluations. We start by motivating the design, followed by an overview of the architecture, and a discussion of various evaluation results that show the promise of the approach for explainable open-world visual modules for Industry 4.0.

2 BACKGROUND AND MOTIVATION

Integrating visual information in automated industrial systems is already widely used. The detection of defective samples is a highly relevant task and is put into an anomaly detection setting by considering defective samples as anomalous. Various deep learning models have shown to detect defective samples [12], but most of them rely on carefully curated datasets and show no insights into their decision making. For sorting tasks, it is mostly assumed that all objects appearing during inference are already known [7] [4] [15] which is not applicable in many realistic scenarios. Although there also exist approaches to sort objects in an unsupervised manner, visual features are extracted mostly based on huge models outputting feature vectors not comprehensible for a human observer [6] or the robot itself is kept in a feedback loop such that it achieves its task accurately at some point [4]. Our Semantic Encoder (Fig. 1) is a convolutional neural network that is trained on purely synthetic image data generated by a renderer, which extracts perceivable visual properties from objects detected in images. Given an image, it outputs a semantic vector where each entry describes a pre-defined property of the depicted object which is intuitively understandable for humans. The concept can be used in four different ways:

Description, classification and characterization. of samples in an explainable manner.
Visual appearances of an object can be described by the Semantic Encoder in a human understandable way. Objects of the same type display similar appearances and will therefore show similar extracted semantic parameters. These semantic feature vectors enable sorting of objects based on cluster analysis in the semantic feature space. As the feature vector assigned to an object is composed of human understandable parameters, we can not only describe single objects, but also characterize entire object classes. We can identify differences and similarities between object classes by determining which of their semantic properties vary to what extent.

Detection of anomalous objects. We can use the extracted features of normal (non-defective) objects and anomalous (defective) objects to distinguish them. Once something anomalous is found, we can pinpoint which semantic parameter is accountable for the deviation from a normal appearance and communicate the difference so that humans understand the deviation intuitively.

Search for objects with specific semantic attributes. Objects with predefined semantic characteristics can be searched for by selecting objects whose semantic descriptions match best with the semantic attributes of the search query.

Generation of predefined "template" objects. The renderer, which is used to generate the synthetic training images based on given semantic attributes (i.e. inverse of the Semantic Encoder), can be utilized to generate "template" objects with explicitly defined appearances.

We use the publicly available MVTech Screws [13] dataset to investigate how the Semantic Encoder could be used in a first simple application, where a system, e.g. a robot, is confronted with objects placed on a surface and the task to describe, sort or find specific objects. We first measure the quality of its extracted feature vectors by comparing them with feature vectors extracted by a VGG-16 model [11] and feature vectors extracted with conventional image processing methods. Our comparison is made based on the Generalized Discriminative Value (GDV) [10], which quantifies the separability of the feature vector clusters by considering how compact and disjoint they are. Next, we investigate our Semantic Encoder in a defect detection setting by quantifying its ability to distinguish between samples from normal and anomalous classes in terms of ROC and PR curves. We also simulate a sorting task, where objects are present from classes, which are labelled, and from classes, that are not labelled at all. We evaluate how accurate class labels are assigned to a set of unlabelled data in which both classes with and without labels are present by using confusion matrices and class assignment accuracy. To investigate the explanatory value of the Semantic Encoder further, we show in a qualitative manner how differences and similarities between entire classes and individual objects can be interpreted easily in a human understandable manner. The geometrical concept we propose demonstrates that a model can be trained to learn parameterizations of an implicit mathematical equation. The extracted semantic properties are mapped onto human understandable descriptions, enabling comprehensible and transparent operations.

Our contribution comprises

- implementation and training details of our Semantic Encoder,
- a quantitative comparison of features extracted by the Semantic Encoder, classical image processing methods and a VGG-16 - based feature extractor, respectively,
- a qualitative and quantitative evaluation of the anomaly detection performance of our Semantic Encoder,
- a qualitative and quantitative evaluation of the Semantic Encoder in a classification-based sorting task,

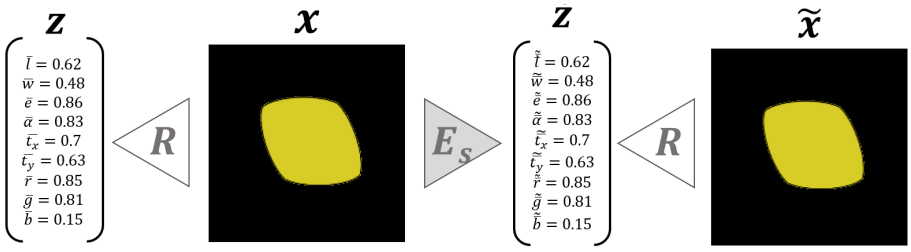


Figure 2: A randomly initialized vector z is rendered by the renderer R to a superellipse image x, which is the input to the Semantic Encoder E_s. The Semantic Encoder is trained to learn the parameters of z, which can be rendered back to an image.

- a procedure outlining the utilization of semantic parameters for transparent and human understandable descriptions of objects.

3 METHODS

3.1 Superellipses for Semantic Encoder Training

In order to train the Semantic Encoder E_s to extract geometric and color information, we utilize "superellipses", also known as Lamé curves. In a Cartesian coordinate system, a superellipse can be defined as the set of points (x, y) which satisfy the following equation:

$$\left|\frac{x}{l}\right|^e + \left|\frac{y}{w}\right|^e = 1, \tag{1}$$

where $e \in \mathbb{R}_0^+$ is a real valued number which describes the "edginess" of the superellipse and $l \in \mathbb{R}_0^+$ and $w \in \mathbb{R}_0^+$ are real valued numbers denoting the half axes. A translation $t_x \in \mathbb{R}$ in x-direction, a translation $t_y \in \mathbb{R}$ in y-direction and a rotation by angle $\alpha \in \mathbb{R}_0^+$, $0 \leq \alpha < \pi$, can be applied by multiplying the set of points which satisfy equation 1 with translation matrix $\mathbf{T} \in \mathbb{R}^{3\times3}$ and rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \mathbf{T}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \tag{2}$$

$$\begin{pmatrix} x'' \\ y'' \\ 1 \end{pmatrix} = \mathbf{R}\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}. \tag{3}$$

We use the normalized parameters of equation 1 together with normalized values of α, t_x and t_y to define a semantic vector $\mathbf{z} = (\bar{l}, \bar{w}, \bar{e}, \bar{\alpha}, \bar{t_x}, \bar{t_y}, \bar{r}, \bar{g}, \bar{b})$. Scalars $\bar{r}, \bar{g}, \bar{b}$ are normalized red-, green- and blue color channel values.

The vector $\mathbf{z} \in \mathbb{R}^{1\times9}$ is initialized randomly and then used by a renderer R to generate the image $\mathbf{x} \in \mathbb{R}^{C\times H\times W}$ depicting the superellipse described by the values in $\mathbf{z}$. C denotes the color channels, H the height and W the width of image $\mathbf{x}$. The renderer R simply takes the parameters in $\mathbf{z}$ as input, plots a coloured superellipse based on equation 1 and applies translation and rotation to it. A convolutional encoder network - the Semantic Encoder E_s - is then trained to learn the parameters $\tilde{\mathbf{z}} = (\tilde{\bar{l}}, \tilde{\bar{w}}, \tilde{\bar{e}}, \tilde{\bar{\alpha}}, \tilde{\bar{t_x}}, \tilde{\bar{t_y}}, \tilde{\bar{r}}, \tilde{\bar{g}}, \tilde{\bar{b}}) \in \mathbb{R}^{1\times9}$ of the depicted superellipse. R can be used to render an image $\tilde{\mathbf{x}} \in \mathbb{R}^{C\times H\times W}$ depicting $\tilde{\mathbf{z}}$, see Eq. 4. The renderer R is a simple forward procedure, whereas the Semantic Encoder E_s constitutes the inversion of that process.

$$\tilde{\mathbf{x}} = R(\tilde{\mathbf{z}}) = R(E_s(\mathbf{x})) = R(E_s(R(\mathbf{z})) \tag{4}$$

3.2 MVTech Screws Dataset

We use the publicly available MVTech Screws [13] dataset to examine the informative value of the extracted semantic information. The dataset consists of 384 top-view images of 13 different screw types lying on a wooden surface. For the evaluation of the quality of the extracted semantic feature vectors (Sec. 3.4), all objects are used. For the anomaly detection experiments (Sec. 3.5) and the classification task (Sec. 3.6), the available objects of each class are split into training, validation and test set with set sizes of 70 %, 15 % and 15 %, respectively. For the anomaly detection experiments (Sec. 3.5), the training set of the known classes is used to determine standard deviation and mean values needed for the computation of the Mahalanobis distance [8], the evaluations are then computed on the validation set of both known and novel classes. For the classification experiments, labeled feature vectors (known classes) are from the validation set and unlabeled features (known and novel classes) are from the test sets of the object classes, see Sec. 3.6 for details.

3.3 Object Segmentation

The screws are placed on a wooden background. In order to extract feature vectors of depicted objects, we first need to segment the objects. We use the GrabCut algorithm [9] for the segmentation. The user input is simulated by annotating the center line of one screw per image using the available bounding box ground truth, see Fig. 4. Each segmented object is then processed individually.

3.4 Evaluation of the Quality of the extracted Semantic Feature Vectors

In order to quantify the quality of the Semantic Encoder descriptions, we compare the features extracted by the **Semantic Encoder** and a **VGG-16 [11] model**, pretrained on ImageNet [3], with features based on **image moments** and **mean color values**. We evaluate the quality of the clustering in terms of the Generalized Discriminative Value (GDV) introduced in [10]. The GDV is computed on a set of vectors and it is defined as the difference between the mean intra-cluster variability and the mean inter-cluster separation. The more compact and mutually disjoint the clusters are, the lower the value. The GDV is −1 in the case

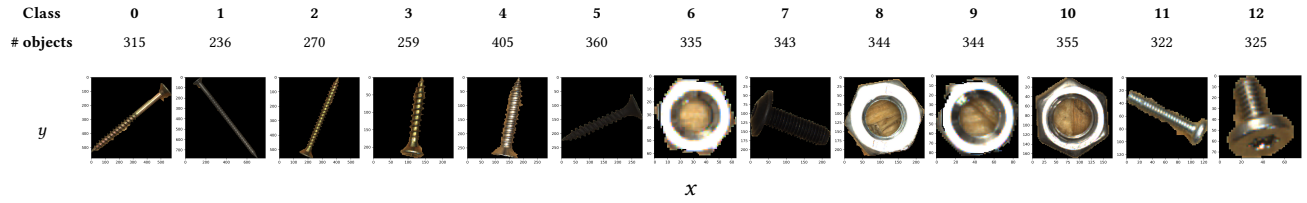


Figure 3: Examples of one segmented object each, for all 13 classes together with the number of objects per class (# objects) used. x and y denote the axis descriptions.

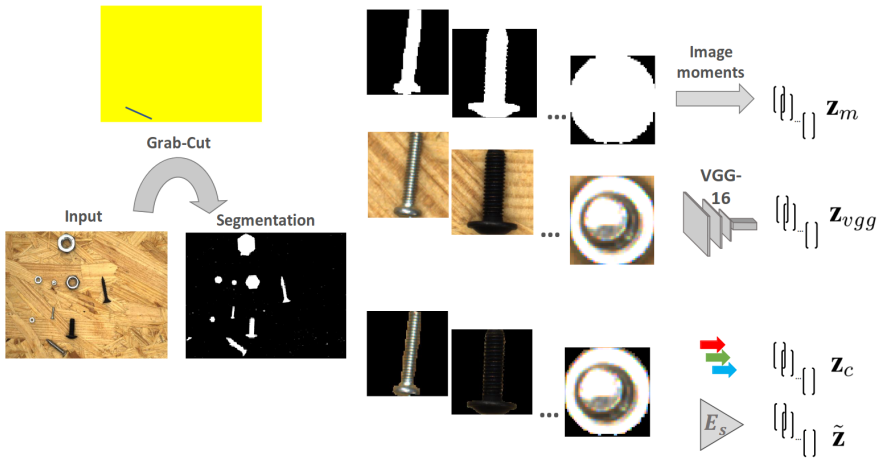


Figure 4: Objects are segmented by using the GrabCut algorithm [9]. Pixels at the center line of one object per image are annotated as obvious foreground pixels (dark blue pixels). All other pixels are treated as possible background pixels (yellow pixels). Image moments are computed on segmentation masks of individual objects. The VGG-16 model [11] gets object image patches as inputs. Masked object patches are used as input to the Semantic Encoder and for mean color value computations.

of perfect separability and 0 for feature points with randomly shuffled labels. To ensure invariance against scale and translation, each feature vector is z-scored along each feature dimension, i.e. we subtract mean and divide by standard deviation as proposed in [10].

We compute feature vectors for each segmented object (Fig. 4):

- Feature vectors $\mathbf{z}_m$ based on **image moments**:
  For a grayscale image with pixel intensity $g(x, y)$ at pixel location (x, y), image moments $M_{i,j}$ of order $i + j$ can be defined as given in Eq. 5.

$$M_{i,j} = \sum_x \sum_y x^i y^j g(x, y) \tag{5}$$

  Central image moments up to third order are computed on the objects' segmentation masks as follows, using centroid coordinates $\overline{x} = \frac{M_{10}}{M_{00}}$ and $\overline{y} = \frac{M_{01}}{M_{00}}$:

$$\mu_{i,j} = \sum_x \sum_y (x - \overline{x})^i (y - \overline{y})^j g(x, y). \tag{6}$$

  We compose feature vectors from central image moments up to third order as $\mathbf{z}_m = (\mu_{00}, \mu_{11}, \mu_{20}, \mu_{02}, \mu_{21}, \mu_{12}, \mu_{30}, \mu_{03})$.

- Feature vectors $\mathbf{z}_c$ based on **mean color values**:
  The mean values of red, green and blue color channel, r_m, g_m and b_m, respectively, of the objects' masked RGB-image are used for the construction of a feature vector $\mathbf{z}_c = (r_m, g_m, b_m)$ for each segmented object.
- Feature vector $\tilde{\mathbf{z}}$ extracted by the **Semantic Encoder**:
  The Semantic Encoder is applied on the objects' masked RGB-image to extract geometrical and color-based descriptions which are represented in feature vector $\tilde{\mathbf{z}} = (\tilde{\tilde{l}}, \tilde{\tilde{w}}, \tilde{\tilde{e}}, \tilde{\tilde{\alpha}}, \tilde{\tilde{t_x}}, \tilde{\tilde{t_y}}, \tilde{\tilde{r}}, \tilde{\tilde{g}}, \tilde{\tilde{b}})$.
- Feature vector $\mathbf{z}_{vgg}$ computed by a **VGG-16 model**:
  A VGG-16 classifier model, pretrained on ImageNet [3], is applied on the objects' RGB-image and the features in the penultimate layer of the model are used as feature vector $\mathbf{z}_{vgg}$ with 4096 parameters for each object.

The geometrical attributes $\tilde{\tilde{l}}, \tilde{\tilde{w}}, \tilde{\tilde{e}}, \tilde{\tilde{\alpha}}, \tilde{\tilde{t_x}}, \tilde{\tilde{t_y}}$ captured by the Semantic Encoder are compared with the geometry information in $\mathbf{z}_m$. To evaluate the accuracy of the color-based attributes $\tilde{\tilde{r}}, \tilde{\tilde{g}}, \tilde{\tilde{b}}$, mean color values in $\mathbf{z}_c$ are used for comparison. Geometrical and color-based information are combined by concatenating feature vectors.

3.4.1 Generalized Discriminative Value (GDV) [10]. To capture the informative power of the different feature vectors, we computed the Generalized Discriminative Value (GDV) [10] which quantifies compactness within and distinction between clusters of feature vectors. We consider a set of N feature vectors in D dimensions $\mathbf{x}_{n=1,...,N}$ and L different classes $C_{a=1,...,A}$. A label $a \in \{1, ..., A\}$ is assigned to each feature vector which indicates the class. Each dimension d is z-scored separately by subtracting the mean $\mu_d = \frac{1}{N}\sum_{n=1}^{N} x_{n,d}$ and dividing by the standard deviation $\sigma_d = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(x_{n,d} - \mu_d)}$ of dimension d to achieve invariance against scaling and translation:

$$s_{n,d} = \frac{1}{2}\frac{x_{n,d} - \mu_d}{\sigma_d}. \tag{7}$$

The z-scored feature vectors $\mathbf{s}_n = (s_{n,1}, ..., s_{n,D})$ are then used to compute mean intra class distance $\overline{d}(C_a)$ and inter class distance $\overline{d}(C_a, C_m)$:

$$\overline{d}(C_a) = \frac{2}{N_a(N_a-1)} \sum_{i=1}^{N_a-1} \sum_{j=i+1}^{N_a} d(\mathbf{s}_i^a, \mathbf{s}_j^a), \tag{8}$$

$$\overline{d}(C_a, C_m) = \frac{1}{N_a N_m} \sum_{i=1}^{N_a} \sum_{j=1}^{N_m} d(\mathbf{s}_i^a, \mathbf{s}_j^m). \tag{9}$$

The number of points in class C_a and class C_m is denoted with N_a and N_m, respectively, and $\mathbf{s}_i^l$ represents the i-th point of class C_a and $\mathbf{s}_j^m$ represents the j-th point of class C_m. The distance between two points $\mathbf{u}$ and $\mathbf{v}$ in D dimensions is computed by $d(\mathbf{u}, \mathbf{v}) = \sqrt{\sum_{d=1}^{D}(a_d - b_d)}$.

The final discrimination value Δ is constructed from mean intra and inter class distances as follows:

$$\Delta = \frac{1}{\sqrt{D}}\frac{1}{A}\left[\sum_{l=1}^{A} \overline{d}(C_a) - \frac{2}{A-1}\sum_{a=1}^{A-1}\sum_{m=a+1}^{A} \overline{d}(C_a, C_m)\right]. \tag{10}$$

The resulting GDV Δ becomes -1 if two clusters of Gaussian distributed points are located such that mean inter cluster distance is two times the standard deviation of the clusters. For unstructured data, the GDV Δ becomes 0 and for very well structured data, the GDV Δ is -1.

For the MVTech Screws [13] dataset, we have $N = 4213$ different feature vectors for each feature extraction method and $A = 13$ different classes. The dimension D varies between different feature extraction methods and denotes the amount of values per feature vector.

3.5 Semantic Feature Vectors for Anomaly Detection

The feature vectors computed for each object are further used for anomaly detection. We assume one class to be anomalous and all other classes as normal. We split the objects from each class into a train and a validation set. We compute the Mahalanobis distance [8] between each feature vector and the set of feature vectors of the training set of normal classes. This results in 12 distance values for each sample in the validation set. We aggregate those distances to a single anomaly score by taking the minimum. The performance is measured by receiver-operating characteristic (ROC) and precision recall (PR) curves, based on TP, TN, FP and FN. TP denotes the amount of objects detected accurately as anomalous, TN denotes the number of objects detected accurately as normal, whereas FP represents the amount of normal objects detected falsely as anomalous and FN denotes the number of anomalous objects not detected as novel. The ROC curve compares true positive rate (TPR = $\frac{TP}{TP+FN}$, aka recall) against true negative rate (TNR = $\frac{TN}{TN+FP}$) and PR curve visualizes the tradeoff between recall and precision ($\frac{TP}{TP+FP}$) [5]. Additionally, we show how the semantic parameters can be interpreted in order to explain why certain anomalous classes are harder or easier to detect. We compute "template feature vectors" for each class cluster by computing the mean values over all feature vectors of the training set of a class. Based on the differences between corresponding parameters between different classes, we can tell which parameters are more similar or more distinct.

3.6 Classification of known and novel classes

We classify known and novel classes by clustering the computed features with a semi-supervised k-means algorithm as proposed in [14] and used by [5]. For all known classes, we provide labels in the validation set. Data points of the test set are unlabelled for known and novel classes. Given the labels of known objects of the validation set, the algorithm first assigns centroids by computing the average of those labelled data points. Starting from these initial centroids, k-means++ [1] is used to obtain the centroids for the novel clusters, for which we do not have any labels. The k-means++ algorithm is the standard k-means algorithm with an improved centroid initialization technique. Having obtained centroids for all classes, a cluster label is assigned to each unlabelled feature vector based on its nearest centroid. The centroids are then updated by averaging all data points of a cluster. The assignment to clusters and the update of centroids is repeated until the k-means algorithm converges. The labelled data points are forced to follow their ground-truth label. We evaluate classification performance in terms of clustering accuracy of unlabelled feature vectors of known and novel classes and confusion matrices. We s-fold cross-validate by randomly shuffling the class names and considering all possible partitions of s consecutive classes as novel and the remaining $(13 - s)$ classes as known, for $s \in [1, ..., 12]$. To get insights, we compute "template feature vectors" for each class cluster by computing mean values over all feature vectors of the training set of a class. These differences reveal why and to what extent classes resemble one another or why they differ.

4 RESULTS

4.1 Evaluation of the Quality of the extracted Semantic Feature Vectors

Tab. 1 shows the clustering performance of different feature vectors. One can see that for the dataset used (see Sec. 3.2), color information plays the most significant role for distinguishing between different object types in terms of GDV, see Sec. 3.4.1 for details about the metric.

The best results are achieved by the feature vectors composed of mean color values, $\mathbf{z}_c$, followed by the clustering performance

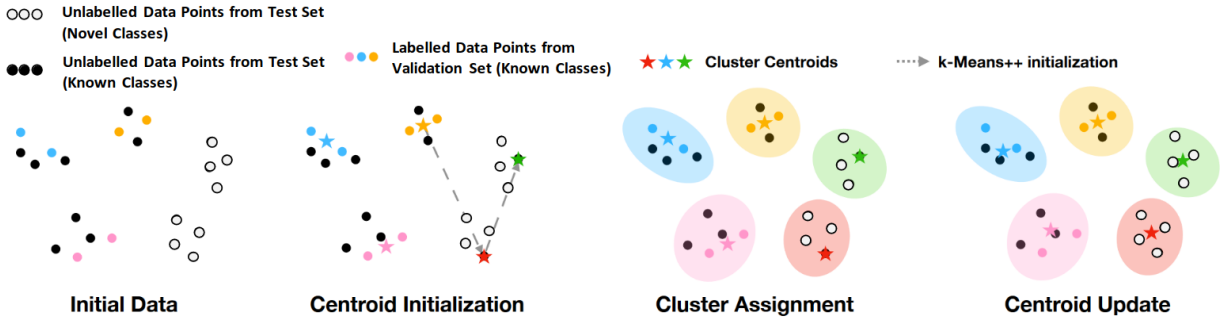


Figure 5: Visualization of the semi-supervised k-means algorithm as proposed in [14]. We have labelled validation data of known classes and unlabelled data from known and novel classes. The semi-supervised k-means algorithm initializes centroids of the labelled (known) clusters based on the average of the labelled data points. Centroids of the unlabelled (novel) clusters are initialized with the k-means++ algorithm. Data points are then assigned to the nearest cluster centroid. Based on the assigned data points, centroids are updated. Image taken and adapted from [14].

Table 1: Generalized Discriminative Value (GDV) [10] results of different combinations of feature vectors. For the Semantic Encoder, we consider either only geometry-based ($\tilde{\tilde{l}}, \tilde{\tilde{w}}, \tilde{\tilde{e}}, \tilde{\tilde{\alpha}}, \tilde{\tilde{t_x}}, \tilde{\tilde{t_y}}$) or only color-based parameters ($\tilde{\tilde{r}}, \tilde{\tilde{g}}, \tilde{\tilde{b}}$) and combine them with features $\mathbf{z}_m$ and $\mathbf{z}_c$ by concatenation. A lower GDV value represents a better performance, see Sec. 3.4.1 for details.

Used Features \ Used Features	No Color	Our Semantic Encoder $\tilde{\tilde{r}}, \tilde{\tilde{g}}, \tilde{\tilde{b}}$	Mean RGB color $\mathbf{z}_c$	Full Color
No geometry		-0.2876	-0.3666	
Our Semantic Encoder ($\tilde{\tilde{l}}, \tilde{\tilde{w}}, \tilde{\tilde{e}}$)	-0.1782	-0.2303	-0.268	
Our Semantic Encoder ($\tilde{\tilde{l}}, \tilde{\tilde{w}}, \tilde{\tilde{e}}, \tilde{\tilde{\alpha}}, \tilde{\tilde{t_x}}, \tilde{\tilde{t_y}}$)	-0.1159	-0.1743	-0.1921	
Central Image Moments $\mathbf{z}_m$	-0.1496	-0.2089	-0.2516	
VGG-16 [11] $\mathbf{z}_{vgg}$				-0.0279

of the color-based parameters $\tilde{\tilde{r}}, \tilde{\tilde{g}}, \tilde{\tilde{b}}$ extracted by our Semantic Encoder. When only geometry is considered, extracted rotation and translation parameters are deteriorating clustering performance, as objects of the same screw type appear with different rotations. When those "disruptive" parameters are removed and only main axes and edginess ($\tilde{\tilde{l}}, \tilde{\tilde{w}}, \tilde{\tilde{e}}$) considered, the clustering performance increases significantly and outperforms image moment clustering. The feature vectors computed by the VGG-16 model show surprisingly poor performance. The GDV value averages over all classes. Plotting the inter-class distances used for GDV computation shows nicely, which parameters lead to good separation between individual classes, see Fig. 6.

Ideally, classes have a large distance to other classes and small distances within themselves resulting in low values on the main diagonal of the plot and higher values anywhere else. The plots reveal that when feature vectors are computed based on color information only ($\mathbf{z}_c$), class clusters which differ a lot with respect to their colors have large distances between them and can therefore be distinguished well. However, for classes with similar colors, geometry information ($\tilde{\tilde{l}}, \tilde{\tilde{w}}, \tilde{\tilde{e}}$) is crucial.

It should be noted further that although rotation and translation do not provide beneficial information for distinguishing between different screw types, they are useful in in other tasks. One application could be to verify if an object's placement on a conveyor belt deviates from the assigned location in a manufacturing scenario.

4.2 Semantic Feature Vectors for Anomaly Detection

ROC and PR curves in Fig. 7 show that some classes, such as 8 or 1, are detected well as anomalous by our method while others, such as 0 or 5, are more challenging. We can examine the depicted difference between "template feature vectors" of the anomalous classes to all normal classes visualized in Fig. 8 to get insights. Every column represents the difference between (anomalous) class 8, 1, 0 or 5 to all other normal classes. The row represents the corresponding semantic parameter. The closer the difference value is to 0, the more similar are two classes w.r.t. that semantic parameter. Differences represented by red tiles indicate that the anomalous class (8, 1, 0 or 5), has a higher semantic parameter value than the normal class we compare it to.

Differences represented by blue tiles indicate that the anomalous class, to which differences are calculated, has a lower semantic parameter value than the normal class we compare to. Considering the differences between semantic parameters reveals why some classes are detected correctly in many cases while objects of other

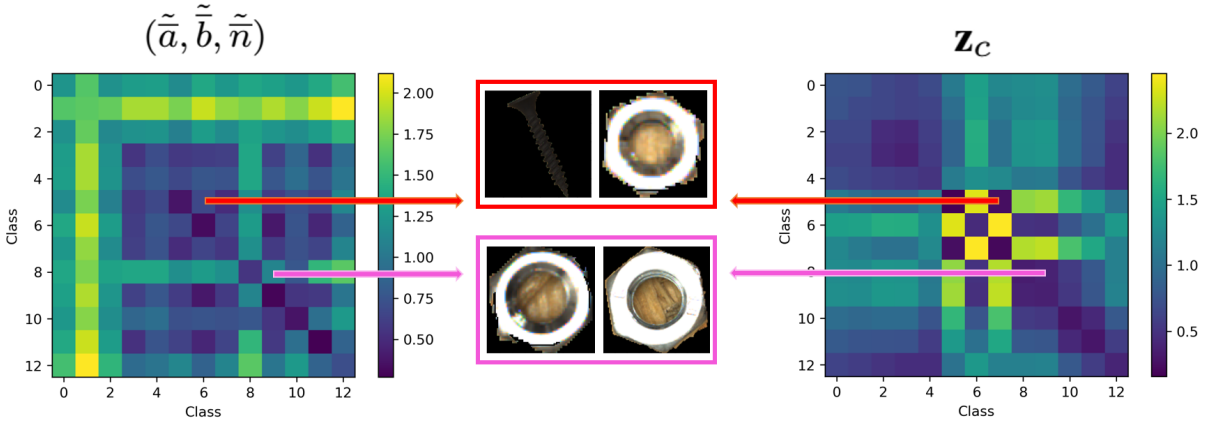


Figure 6: Inter-class distances used for GDV computation. Ideally, values below/ above the main diagonal are significantly higher than values on the main diagonal. When only geometry is used ($\tilde{\bar{l}}, \tilde{\bar{w}}, \tilde{\bar{e}}$), classes which differ a lot in terms of length or width, are showing higher inter-class distances between them. Relying on color-based information only (z_c) leads to high inter-class distances for classes which differ a lot w.r.t. their colors.

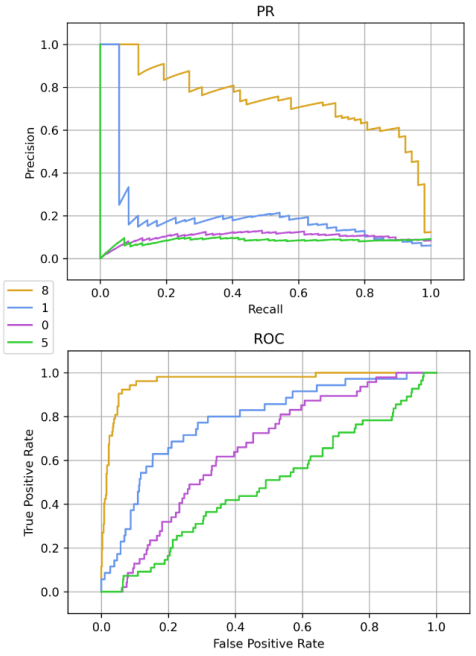


Figure 7: ROC and PR curves represent anomaly detection performance. For each graph, one class is considered to be novel and all other classes are considered to be known.

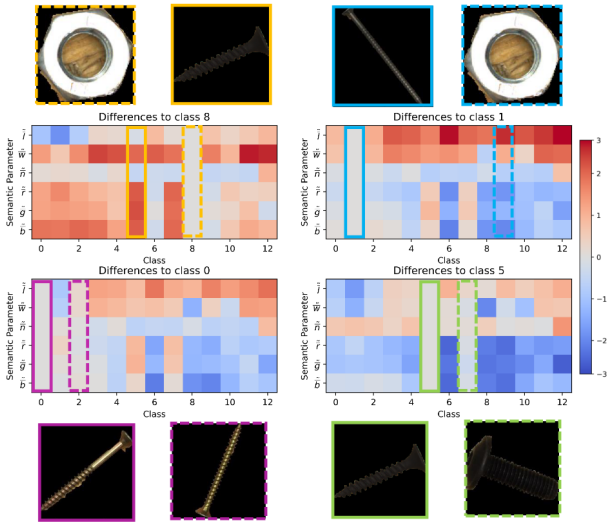


Figure 8: Differences between "template feature vectors" of class 8, 1, 0 and class 5 to templates of all other classes. Red boxes show that the difference was positive - meaning that the corresponding parameter value of class 8, 1, 0 or 5 is higher than than the compared value. Blue boxes indicate that the value of class 8, 1, 0 or 5 is lower than the value of the class we compare to.

classes are confused with normal object types. Class 8 and 1 differ "enough" from all other classes while class 5 and 0 are "too" similar to classes 7 and 2, respectively, to be distinguished well. The upper left plot in Fig. 8 shows that objects of class 8, which was detected well as anomalous, differs strongly from other classes w. r. t. all semantic parameters. For class 1 (upper right plot), which was detected rather accurate too, differences to all normal classes are relatively pronounced too. We can see why the detection of class 0 and 5 resulted in worse performance. Objects of anomalous class 0 are very similar to objects of normal class 2 and objects of anomalous class 5 show strong similarities to normal class 7, see lower left and lower right plot in Fig. 8, respectively. We can also reveal similarities or dissimilarities of semantic parameters. As an example, according to $\tilde{\bar{r}}, \tilde{\bar{g}}, \tilde{\bar{b}}$ in the upper left plot in Fig. 8, class 8

has significantly higher color values than class 6. Its width, represented by parameter $\tilde{\tilde{w}}$, is higher and its length $\tilde{\tilde{l}}$ slightly smaller. This can be verified by looking at Fig. 3. Class 1 (upper right plot in Fig. 8), shows darker $\tilde{\tilde{r}}, \tilde{\tilde{g}}, \tilde{\tilde{b}}$ values than class 9, while having greater length $\tilde{\tilde{l}}$ and slightly broader width $\tilde{\tilde{w}}$. This can be seen by looking at Fig. 3 too. Edginess parameters $\tilde{\tilde{e}}$ are less comprehensible, we suspect that the segmentation boundaries are too inexact to compute reasonable parameters for the objects. Moreover, all screws are very similar in that regard anyway.

4.3 Classification of known and novel classes

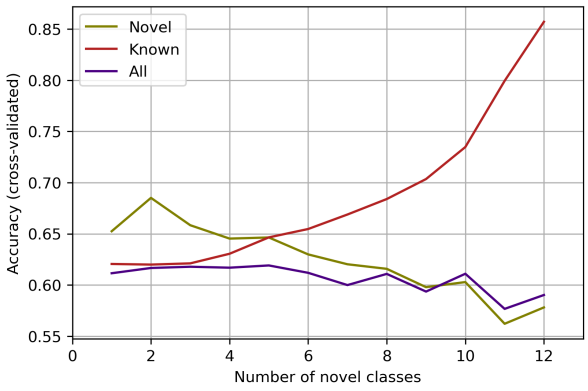
Classification results are shown in Fig. 9. Clustering accuracy for novel objects decreases for higher numbers of novel classes. Clustering for all unlabelled data points, from known and novel classes, decreases too. This is reasonable because for novel classes, no labels are available. The higher the number of novel classes, the higher the fraction of classes for which we have no available labels, resulting in a more challenging task. Clustering accuracy for known classes increases for higher numbers of novel classes. This is due to the fact that clustering for known classes gets easier if less classes are known because a larger fraction of datapoints corresponds to novel classes. The confusion matrices in Fig. 9b shows that objects of class 7 are assigned to class 5 in all cases. Objects of class 6 are confused with objects of class 9, vice versa objects of class 9 are assigned to class 6. By looking at the objects in Fig. 3, the miss-classifications become plausible - there is a strong similarity between class 5 and 7 with respect to their shape and color, the same holds for class 6 and 9. The semantic parameters represent these findings too, as can be seen in Fig. 9c, where we plotted differences between "feature template vectors" of class 7 and 9 to all other classes.

5 CONCLUSIONS

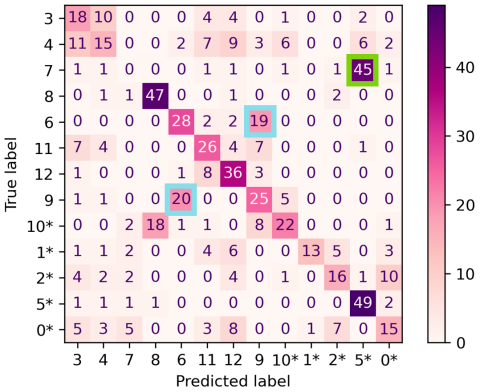
We proposed to incorporate semantic information into a comprehensible feature vector description of objects. The computed feature vectors can be used to describe objects, detect anomalies and sort objects by classifying them.

We used the publicly available MVTech Screws [13] dataset to present how the concept can be utilized in an industrial AI-based system. We measured the informative value of the extracted semantic representation and evaluated the concept in an anomaly detection and a classification-based sorting task, showing how the results can be interpreted in an easy human understandable manner.

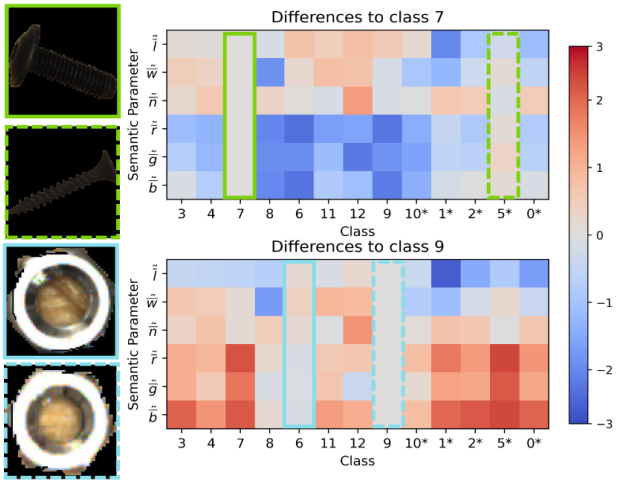
The semantic features we used for the presented tasks were extracted without any retraining on the evaluation dataset. Rather, the Semantic Encoder was only trained on a simple, synthetic dataset, still allowing the trained model to process real world objects. Using only a very small set of parameters, we were able to construct explainable feature vectors that are more informative than a large VGG-16 model trained on large amounts of data. While anomaly detection performance was good for some classes, objects of other classes are often misclassified. However, we can see that these misclassifications are comprehensible—some classes were indeed very similar to each other and we can pinpoint which semantic properties lead to the wrong assignment. This shows the strength of the Semantic Encoder concept in terms of explainability—we can represent the extracted feature vectors of a class in a way such that we can understand which parameters led to poor performance.



(a) Clustering accuracy for unlabelled novel, known and all (novel and known) objects.



(b) Confusion matrix showing classification performance of one partition of size $s = 3$.



(c) Differences between "template feature vectors" of class 7 and 9.

Figure 9: Classification performance in terms of clustering accuracy, one exemplary confusion matrix and differences between 'template feature vectors", which can be used to reveal additional findings.

We show that a machine learning model is able to learn the parametrization of a defined mathematical formulation, enabling comprehensible descriptions of objects. We use geometry-based superellipse equations, but the concept can be further expanded to a wide range of implicit mathematical formulations. The trained model does not need to be retrained on the data at hand, thus being able to cope with realistic open-world settings. The experiments presented in this work are a first conceptual approach towards an explainable AI system, capable to handle several tasks in industrial settings in a transparent manner, bringing us one step closer towards trustworthy and explainable AI-supported industry. Describing objects with one superellipse only is of course limited in its informative power and leads to mistakes in the anomaly detection and the classification task. In ongoing work, we aim to improve our concept by partitioning an object into its individual object parts. A description of the object's parts is then generated and combined with a description of the spatial relationship of the parts to each other. We expect these more extensive descriptions to capture differences and similarities of objects better leading to an increase in anomaly detection and classification performance. Furthermore, we intend to map the semantic parameters to human language to generate an effortless interpretation and we intend to expand the semantic descriptions with more complex geometrical shapes.

REFERENCES

[1] David Arthur and Sergei Vassilvitskii. 2007. K-Means++: The Advantages of Careful Seeding *(SODA '07)*. Society for Industrial and Applied Mathematics, USA, 1027–1035.

[2] Terrance Boult, S. Cruz, Akshay Dhamija, Manuel Günther, James Henrydoss, and W.J. Scheirer. 2019. Learning and the Unknown: Surveying Steps toward Open World Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (07 2019), 9801–9807. https://doi.org/10.1609/aaai.v33i01.33019801

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[4] Patrik Fager, Robin Hanson, Åsa Fasth-Berglund, and Sven Ekered. 2021. Supervised and unsupervised learning in vision-guided robotic bin picking applications for mixed-model assembly. *Procedia CIRP* 104 (2021), 1304–1309. https://doi.org/10.1016/j.procir.2021.11.219 54th CIRP CMS 2021 - Towards Digitalized Manufacturing 4.0.

[5] Patrick Feeney, Sarah Schneider, Panagiotis Lymperopoulos, Liping Liu, Matthias Scheutz, and Michael C. Hughes. 2022. NovelCraft: A Dataset for Novelty Detection and Discovery in Open Worlds. https://doi.org/10.48550/ARXIV.2206.11736

[6] Joris Guérin, Thiery Stéphane, Eric Nyiri, and Olivier Gibaru. 2018. Unsupervised Robotic Sorting : Towards Autonomous Decision Making Robots. *International Journal of Artificial Intelligence and Applications* 9 (03 2018), 81–98. https://doi.org/10.5121/ijaia.2018.9207

[7] Maria Koskinopoulou, Fredy Raptopoulos, George D. Papadopoulos, Nikitas Mavrakis, and Michail Maniadakis. 2021. Robotic Waste Sorting Technology: Toward a Vision-Based Categorization System for the Industrial Robotic Separation of Recyclable Waste. *IEEE Robotics & Automation Magazine* 28 (2021), 50–60.

[8] Prasanta Chandra Mahalanobis. 1936. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)* 2 (1936), 49–55.

[9] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. 2004. "GrabCut": Interactive Foreground Extraction Using Iterated Graph Cuts *(SIGGRAPH '04)*. Association for Computing Machinery, New York, NY, USA, 309–314. https://doi.org/10.1145/1186562.1015720

[10] Achim Schilling, Jonas Rietsch, Richard Gerum, Holger Schulze, Claus Metzner, and Patrick Krauss. 2018. How deep is deep enough? - Optimizing deep neural network architecture. (11 2018).

[11] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*.

[12] Andrei-Alexandru Tulbure, Adrian-Alexandru Tulbure, and Eva-Henrietta Dulf. 2022. A review on modern defect detection models using DCNNs – Deep convolutional neural networks. *Journal of Advanced Research* 35 (2022), 33–48. https://doi.org/10.1016/j.jare.2021.03.015

[13] Markus Ulrich, Patrick Follmann, and Jan-Hendrik Neudeck. 2019. A comparison of shape-based matching with deep-learning-based object detection. *tm - Technisches Messen* 86, 11 (2019), 685–698. https://doi.org/10.1515/teme-2019-0076

[14] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. 2022. Generalized Category Discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 7492–7501.

[15] Yawan Zhang and Wei Cheng. 2019. Vision-based robot sorting system. *IOP Conference Series: Materials Science and Engineering* 592 (09 2019), 012154. https://doi.org/10.1088/1757-899X/592/1/012154