EC3

Methodological Guideline for Anomaly Detection Models

L3.5.2.3

contact@confiance-ai.fr | www.confiance.ai

Document reference: 321FA

Contributors

	Name	Organisation	Role
Responsible for the deliverable	Héléna Vorobieva	Safran	Data Scientist
Scientific responsible	Héléna Vorobieva	Safran	Data Scientist
Co-authors	Héléna Vorobieva	Safran	Data Scientist

Document Control

Revision	Date	Commentary	Author
v0.1	2023/12/05	Extract from batches 1 and 2	Héléna Vorobieva
v1.0	2023/12/15	Release on CAIMAN	Héléna Vorobieva

Contents

A. Introduction and abstract

A.1. General introduction to trustworthy AI challenges

Trustworthiness in AI within critical systems (systems that can directly or indirectly affect human life and moral entities) is essential for its widespread adoption (by the industry, the decision makers, the general public, etc.) and poses the following significant challenges.

- First, how to design AI models, so that, by construction, they satisfy trustworthy properties (accuracy, robustness. . .).
- Secondly, how to characterize these AI models, for example to understand and explain their behavior and their adequacy to the operational domain.
- Then, how to implement and embed those AI models on hardware, by making them fit for the target without losing their trustworthy properties.
- Another question is, what methods of data engineering to apply in order to, among other topics, manage important volumes of data and adapt to the evolution of the operational domain.
- At system level, what verification and certification processes to consider specifically for AI-based systems.
- Finally, a federation of all these matters is necessary to build an end-to-end methodological approach, supported by a consistent engineering environment compatible with industrial practices.

These are the challenges, among others, that the Confiance.ai program addresses.

A.2. Introduction to anomaly detection models

This section introduces briefly notions described in the Project EC3 state of the art: monitoring. This section presents also the typology of studied methods.

A.2.1 Monitoring

Nowadays, most of industry uses technology to a greater or lesser extent. It means that technological services are directly offered to their customers. In most of the cases, a key factor for a business to continue operating is to assure the good functioning of its equipment, networks and systems.

Technology is essential for companies but it can also fail. Many faults of different kinds can occur which may lead to critical situations. For this reason, any company that relies in computer infrastructure needs to control and assure the correct operation of it. The main goal is to assure safety, security and reach performance objectives related to failure detection.

Counting with a good monitoring tool (called from now “monitoring system”) is key for efficiently detect and prevent from possible failures. Monitoring systems are responsible for controlling the systems technology, such as networks and communications, operating systems or applications, etc. They analyse their operation and performance, and raise alerts when errors are detected. Therefore, a proper monitoring system must be able to monitor applications, services, devices or infrastructures among others.

One of the artefact to monitor is anomaly.

A.2.2 Definition of the different classes of anomaly

Anomaly detection is a research area that studies the detection of unusual observations by leveraging methods, models, and algorithms based on data. This field has a long standing history spanning across different domains and applications. For defining anomaly we consider: An anomaly is an observation that deviates considerably from some concept of normality. This definition covers a broad range of use-cases leading to some subtly distinct classes of anomalies, that usually reflect different objectives in an applications. We enumerate here the main instances of anomalies for a better overview and understanding.
The main classes of anomaly and corresponding task objectives are the following:

- outlier detection: data points from the training set that are far from the others, i.e., an unusual or noisy training sample.
- novelty detection: test points that could be new observations, i.e., the equivalent of an outlier for the test data.
- out-of-distribution (OOD) detection: objects that are drawn from a distribution different from the training distribution. In deep learning, we can distinguish two types of OOD: low-level (different pixel statistics due to a mismatch between training and testing environments), high-level (semantic, unknown objects or entities in a familiar environment).
- misclassified samples: objects that are likely to be misclassified and that fall near the decision boundary where the classifier is uncertain.
- anomaly: generic term describing a broad range of anomalous patterns in the training or testing data whatever the cause of irregularity w.r.t. other points. It can encompass OOD, but also unseen instances of know classes. Furthermore it can encompass sometimes aleatoric uncertainty.

We note that in practice and throughout this report the different classes of anomaly are used interchangeably for the sake of simplicity of the text. In practice, the underlying methods display several similarities (some addressing multiple classes at once) and could be repurposed to a different class of anomaly with relatively few changes.

A.2.3 Out of Distribution Detection

A.2.3.1 Rationale

While machine learning (ML) methods have successfully tackled complex real-world problems in various domains (especially deep neural networks DNN), they still face significant limitations that make them unfit for safety-critical applications. A major well-known drawback of ML components consists in their inability to distinguish supported (valid) from unsupported (invalid) inputs. That means that ML models will produce an output even if such input is meaningless in the domain where the model is supposed to operate. This situation results often in erroneous predictions with high confidence, especially in the case of DNNs. Thus, OOD detection techniques are used to avoid the processing of unsupported inputs, which may cause unpredictable predictions, potentially causing severe system-level failures, if not handled properly.
Unsupported inputs that can occur in real-world situations can be categorized into three classes based on their proximity to the training set data distribution. Supported inputs pertain to the same data distribution of the ones used for training (in-distribution inputs) and therefore they should be handled correctly by the DNN. Unsupported inputs, on the other hand, pertain to a different data distribution than the one used for training (out-of-distribution inputs). Underrepresented inputs

is a first category of inputs that can cause a failure of the monitor in learning meaningful patterns for such data, due to their low frequency of occurrence, as compared to other, more represented, classes. A second category consists of novel data, i.e. data in the domain of validity of what the DNN should support, but which are not yet represented at all in the set used for training and should therefore be added to it as representative of new classes of data. Finally, the third category of unsupported inputs that can be found in real-world domains consists of anomalous data, i.e., inputs that are not in the validity domain and, as such, should be recognized and discarded.
Different classes described as anomaly detection in the previous section can also be generalized as different classes of unsupported inputs for ML methods.
In this document, we focus on data driven OOD or anomaly detection.

A.2.3.2 Data-Driven Out-of-Distribution Detection

According to the number of available types of samples (normal, anomalous, and unlabeled) in the training set, OOD or anomaly detection methods can be categorized into three types: Supervised, semi-supervised, and unsupervised detection methods.
Supervised OOoD detection assumes that a well defined explanation of the abnormality exists. Therefore, collecting normal and anomalous samples for training a binary classifier is straightforward. Although supervised approaches produce highly accurate results, their outcomes are not sufficiently generalized. The performance of deep supervised classifier is usually sub-optimal due to the class imbalance.
In the Semi-supervised case, there are numerous unlabeled data in the OoD tasks while collecting anomalous instances is a burdensome and expensive process. This problem emerges because abnormalities are very diverse and rarely occur.
Finally, in the unsupervised case, the OOD task is applied to unlabeled data by training a model which is able to detect outliers based on intrinsic properties of data instances. The assumption in this category is that abnormal events rarely occur in unlabeled samples, just like in realistic situations. In the spectrum of unsupervised anomaly detection solutions, using a reconstruction objective is one of the most common and oldest approach for leveraging (deep) neural networks for anomaly detection. Reconstruction-based methods learn a model that can reconstruct accurately in distribution data points and reconstructs poorly unusual data points. OOD samples are expected to lead to erroneous and less reliable reconstructions as they contain unseen patterns during training.
In this document, we first focus on a new method for improving the detection of OOD (classification or semantic segmentation). This method uses any OOD neural network of classification or segmentation. Then we focus on several reconstruction based OOD detection methods.

A.3. Addressed Confiance.ai scientific challenges

This document addresses the scientific challenge "Components with integrated self-monitoring of the detection of the exit from the operating zone".

A.4. Trustworthiness Attributes

The addressed trustworthiness attributes are reliability and robustness.

A.5. Target audience

This document is targeted to IA engineers who have at least basic knowledge of deep neural networks and who want to decide which method to use. This document has also to be read before trying to implement themselves presented methods, or before using code from demonstrators.

A.6. Document organization and abstract

This report is structured as follows:

- The part A is this introduction.
- The part B presents a new design method for improving the detection of OOD (classification or semantic segmentation). This method can be applied to both non-OOD and OOD anomaly detection. This method can be easily extended to other type of OOD detection. It improves detection as long as a neural network is used to solve this task, and as long as patches of images (instead of full images) have to be the input of the neural network. This method uses ensemble methods with fusion of results from a neural network taken at different convergence points. The added value is how to select the convergence points.
- The part C presents an OOD detection method using Normalizing Flows (NF) developed by the DEEL program. This method can separate OOD and ID data for only if there a no big variations of the seen object to inspect or if the OOD and ID data is very different.
- The part D is dedicated to OOD detection by reconstruction methods with Transformers. Two methods from state of the art have been successfully tested on Safran Visual Industrial Control and Renault Welding use cases. We present these methods and how to use them. The application on the use cases are described in the use case level document. These methods should be used for detection of OOD of type added noise or superimposition, variation of outside conditions. They can also detect anomalies but only if there a no big variations of the seen object to inspect.
- The part E contains a brief state of the art study for OOD detection with diffusion models.
- The part F contains the conclusion and the outlook of this work.

A.7. Acronyms

Acronym	Definition
DNN	Deep Neural Network
CNN	Convolutional Neural Network
ID	In Distribution
ML	Machine Learning
MLP	Multi Layer Perceptron
NF	Normalizing Flows
NN	Neural Network
OOD	Out of Distribution

B. Design method for improving the detection of OOD

In this chapter, a design method for improving the detection of OOD of type anomaly is described. This method can be applied to both non-OOD and OOD anomaly detection. This method can be easily extended to other type of OOD detection.
This method was created for the Safran Visual Industrial Control use case, i.e. in the context of automatic or semi-automatic non-destructive control of an aeronautical part based on one or more images of the part. It can be extended to any inspection task, as long as a neural network is used to solve this task, and as long as patches of images (instead of full images) have to be the input of the neural network.
This method takes as input a neural network of classification or semantic segmentation giving good, but not good enough result on the inspection task or on the OOD of type anomaly detection task. Previous knowledge tells us that consolidations, such as using different convergence epochs (as explained in the following sections) improve the results.

B.1. State of the art

The automatic control of parts by neural network is done in the state of the art by using a single network which calculates a prediction to know if the part is defectuous or not Morard and Parra (2017), Morard (2018).
In order to gain in robustness, the use of several networks via ensemble methods is an approach known to the deep learning community Gal and Ghahramani (2016), Lakshminarayanan et al. (2017). The classification from the obtained networks is classically done by calculating the average of the predictions, by voting or by more advanced techniques Cheng et al. (2018). In the patent Chapdeleine and Picard (2020), the improvement of robustness is obtained by using a set of networks whose good coverage of the parameter space is guaranteed. When several neural networks are used, they can have different architectures, or the same architecture but a different initialization.
Another way to gain robustness and improve results is to use ensemble methods on a single neural network but taken at different convergence points (training epoch number). This way of proceeding has the advantage of training only one neural network. Thus, Cheng et al. (2018) randomly selects 4 different epochs to apply ensemble methods. In a more relevant way, Chen et al. (2017) chooses the best epochs according to the cost function used for the training of the neural network. It should be noticed that this ensemble method can be used alone or as a complement to ensemble methods with separately trained neural networks.

B.2. Objective

In our problem, we consider that there are images to inspect, however, the used deep neural network cannot handle full images and has to make predictions on patches.
Two strategies are then possible to find the anomalies. Either a classification strategy (indicating whether the inspected patch contains an anomaly, as with the neural network provided to Confiance.ai), or a semantic segmentation strategy (indicating on each patch the pixels included in an area containing an anomaly, which is similar to a pixel-by-pixel classification). To train

a neural network to perform either classification or semantic segmentation, cost functions are used. They penalise a bad response: misclassified patch or pixel. In order to improve the results, it is interesting to use ensemble methods. Various ensemble methods can then be used alone or in combination with each other, or in combination with other methods for improving the results. As mentioned at the beginning of this section, in this study we place ourselves in the case of ensemble methods with the training of a single neural network, but taken at different epochs.
The technical problem studied is: how to select the epochs for these ensemble methods. In Chen et al. (2017) the authors choose the best epochs according to the cost function used for training the neural network. Classically, there are other indicators in addition to the cost function to measure the performance of detection or semantic segmentation, for example :

- Mean Intersection over Union (mIoU): The area of overlap of the predicted class with the ground truth class over the total area of the union of the two.
- Accuracy: The percentage of pixels of the class correctly predicted. It is defined as the ratio of true positives TP to the sum of true positives and false positives FP.
- Recall: The percentage of pixels in the class found, out of all pixels in the class. It is defined as the ratio of true positives to the sum of true positives and false negatives.

To our knowledge these indicators are not used in the state of the art for the selection of epochs in ensemble methods.
Whether it is the cost function or the classic indicators mentioned above, these measures do not answer the general problem. Indeed, these measures only favour a good response in relation to the raw ground truth (healthy or anomalous patch or pixel). However, in relation to our general problem, we are looking at another level of precision: we want to know the approximate location of anomalies, with few false alarms. Thus, the classic measurements will be very penalizing if, for example, only a part of the pixels of an anomalous area are classified as anomalies, whereas for the general problem it is correct. Similarly, if a large area of anomalies is found on several patches, it would be sufficient to classify only some of these patches as defects, whereas the measurements will be penalising for the other poorly classified patches in this area. Symmetrically for healthy areas, it is identical from the point of view of the general problem to return an aggregation of misclassified healthy pixels or patches, whatever the size of the aggregation, whereas the measurements will be more penalizing for larger aggregations.
Thus, the state-of-the-art measures are not suitable for selecting the best epochs for our problem. We therefore propose in this study a new score for selecting the best epochs.

B.3. Inputs and outputs

This method requires as input :

- A dataset of images divided into patches for training, validation and testing.
- A nomenclature to reconstruct the images from the patches.
- Annotations for the patches, either in pixel form (semantic segmentation) or with a label for the whole patch (classification).
- The output results of the neural network considered for each training epoch on each patch of the validation set.

This method produces as output :

- A ranking of the epochs according to the score created in this method. The end user, depending on his resource and time constraints, will be able to define the number of best epochs he wishes to use.

B.4. Monitoring class

This is a design method to improve the results of a neural network. This design method does not change the learning strategy (same network, same cost function, same meta-parameters), but proposes to make the final decision from several epochs, chosen according to a new score presented here, instead of making the decision from the only epoch of convergence according to the cost function. This method can be used either alone or to monitor the initial neural network results (taken at the epoch of convergence of the cost function). In this case, a possible strategy could be to filter out discrepancies in the results of

B.5. Design principles

For each training epoch, the patches of the validation set are tested to obtain a result in the form of a patch.These result patches are assembled in order to reconstruct the initial images. These result images are binarised for a plurality of thresholds. For each binarised image, we consider the superposition of the predicted anomaly polygons with respect to the ground truth anomalies. Thanks to the different thresholds, we obtain for each epoch, a curve of anomalies found according to the false alarms. After having been made comparable, we calculate the area under the curves for each epoch, which gives our new score. The epochs are then ranked according to this score and the best N epochs are used in the test.the two methods.

B.6. Detailed design

We describe here in detail how to apply the method for semantic segmentation networks (calibration of the network and usage in test) and specify at the end what are the few modifications to be done for a classification neural network. The proposed method works whatever the semantic segmentation neural network and whatever the associated semantic segmentation cost function.

B.6.1 Calibration

The overall view of the calibration process is given in FigureB.3. The steps that are added compared to a classical neural network calibration are framed in yellow. Thus the calibration blades are divided into training blades and validation blades. In both cases, the blade images are subdivided into patches in module 1 (see FigureB.1). Then these patches are provided to the neural network for training and validation in module 2. While usually this step is sufficient to obtain a trained neural network, we produce additional outputs in module 2 which are used to create the best epoch selection score. Note: Module 2 is not entirely new as we will see later in this section, but for clarity of illustration we have circled it in yellow anyway.

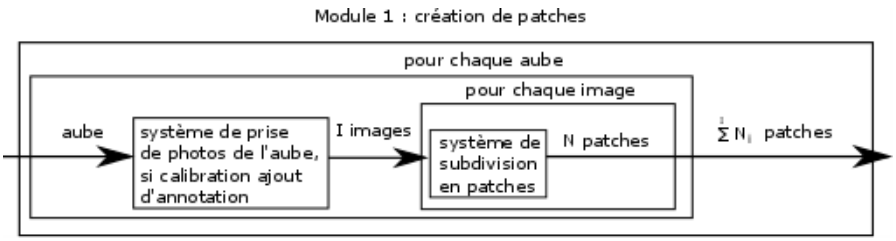

Figure B.1: Patch subdivision module

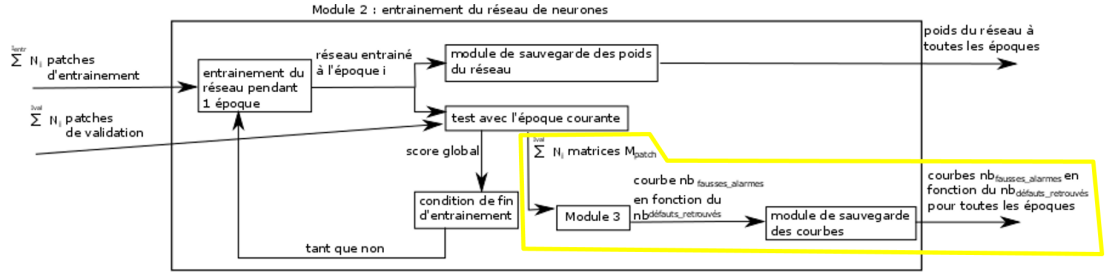

Figure B.2: View of the neural network training sub-process, including in Module 3 the step of calculating the curves allowing the subsequent calculation of the performance measure (score) proposed in this work. The steps added by our proposal are circled in yellow.

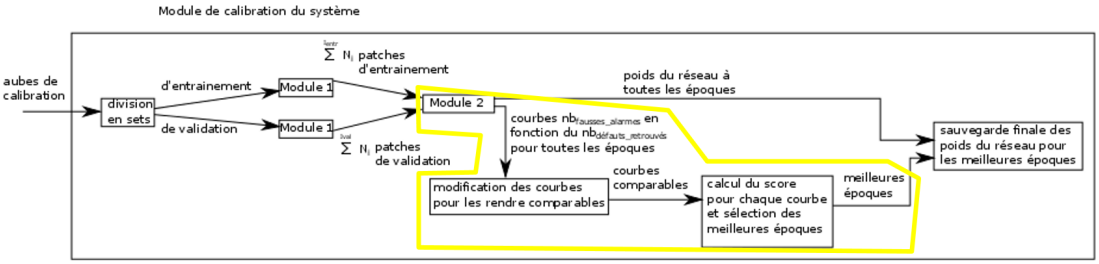

Figure B.3: Overview of the system calibration process, including the step of calculating the performance measure (score). The steps we have added are circled in yellow.

As the training progresses, at the end of each epoch, the new performance measure proposed in this work and explained in the steps below is calculated on the validation set. These steps correspond to the parts circled in yellow in FigureB.3 and FigureB.2.
Steps 1) and 2) explained below correspond to module 3 introduced in FigureB.2 and showed more in detail in FigureB.4. As for steps 3) and 4), they are already present schematically in FigureB.3.

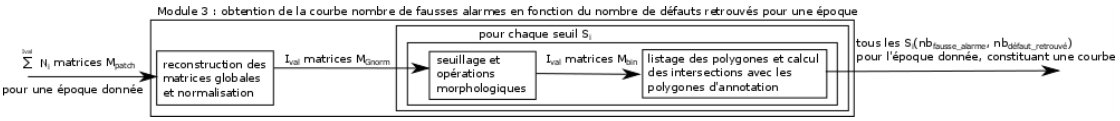

Figure B.4: View of the sub-process of obtaining the curves allowing the performance measure (score) to be calculated afterwards.

B.6.1.1 Construction of a global result matrix for each image of the validation set

For each image of the validation set, we test all the patches and we thus obtain as many small result matrices M_{patch} of the size of the patches. As we know the position of the patches in the image, we construct a global result matrix M_G of the image size by placing the small matrices M_{patch} at the corresponding coordinates of the patches.
In the case where the patches have been created with an overlay, we can have several results for some coordinates. In this case, the results must be merged. One way to proceed is: for the coordinates where we have several results, we add these results in M_G. In parallel, we create a matrix $M_{contribution}$ containing for each coordinate, the number of small matrices having contributed to

the score at this location. For example, if a given pixel is only seen in one patch, the number will be 1 at the corresponding coordinate. If with overlays the pixel is present in N patches, this number will be N. Then, when all the patches of the image have been tested and integrated in M_G and their contribution in $M_{contribution}$, we normalise M_G: $M_{Gnorm} = M_G / M_{contribution}$.
Remarks :

- If there is no overlap of patches, $M_G = M_{Gnorm}$.
- M_{Gnorm} thus contains values between 0 and 1 and can thus be assimilated to a greyscale image (0 is black and 1 is white).

When the whole validation set is tested, we obtain as many M_{Gnorm} as there are images in the validation set.

B.6.1.2 Thresholding and classification of detected anomalies polygons

We fix S thresholds regularly spaced between 0 and 1 (for example from 0.1 to 0.9 by steps of 0.1). The greater the number of thresholds, the more refined the results can be, but the longer the calculation time. Thus, the number of thresholds must be set according to the targeted calculation time.
The following is to be done for each threshold S_i.
Each M_{Gnorm} is thresholded (if the value is strictly greater than the threshold then it is replaced by 1 and otherwise it is replaced by 0) and thus produces binary images M_{bin}. Optionally, morphological opening and/or closing operations can be performed in order to smooth the results or to eliminate small black areas. In these binary images, the black pixels correspond to defect pixels found by the "neural network + M_{Gnorm} construction + thresholding" process.
We list all the black polygons (connectedness 4) P_{result} in M_{bin} whatever their size. We then look at whether or not these polygons intersect annotation polygons P_{annot} :

- For each P_{annot}, we look if there is at least one P_{result} polygon having a non null intersection with this P_{annot}. If this is the case, we consider that the P_{annot} anomaly is found.
- For each P_{result}, we look if there is at least one P_{annot} having a non null intersection with this P_{result}. If this is not the case, we consider that the polygon P_{result} is a false alarm.

This operation being done for all the polygons P_{result} and P_{annot} of all the images of the validation set, we have a couple number of anomalies found and number of false alarms, for each threshold S_i.

B.6.1.3 Obtain comparable curves

At the end of the previous step, we can plot for each epoch the curve of the number of anomalies found as a function of the false alarms (each point of the curve corresponding to a different threshold S, we connect the points in the decreasing order of the thresholds). We note these points $S_i(nb_{falsealarm}, nb_{anomaliesfound})$.
It is then necessary to check that these curves respect some rules. The closer the threshold is to 0, the greater the number of anomalies found must be and the more false alarms we must see. Thus, for 2 given thresholds S_1 and S_2, if S_1 is smaller than S_2, then (assumption 1) the number of anomalies found for S_1 is greater than or equal to the number of anomalies found with S_2 and (assumption 2) the number of false alarms for S_1 is greater than or equal to the number of false alarms with S_2. Mathematically, (assumption 1) is always respected. However, for too low thresholds, (assumption 2) is no longer respected because instead of having many small false alarm zones, we end up with few false alarm zones but which are very extensive. We thus find the list of the points not respecting (hypothesis 2) noted S_h. For these points, the value of the

number of anomalies found and the value of the number of false alarms must be modified. Let $S_N(nb_{falsealarmSN}, nb_{anomaliesfoundSN})$ be the first threshold from which (hypothesis 2) is respected for the considered period. Let F be the maximum number of false alarms over all epochs among the thresholds respecting (hypothesis 2). Then for the considered epoch, we modify the abscissa and ordinate of S_h such that: $S_h(nbfalsealarm, nb_{anomaliesfound}) = (F, nb_{anomaliesfoundSN})$.
This is to be done for all epochs (thus for all curves).
Thanks to this step, the maximum abscissa for all curves is the same. In order to make the minimum abscissa the same for all curves, for the curves where there is no point with abscissa 0, a point $S_0(0,0)$ is added.
This gives curves with the same abscissa values, so that they are comparable.
An illustration is given in FigureB.5.

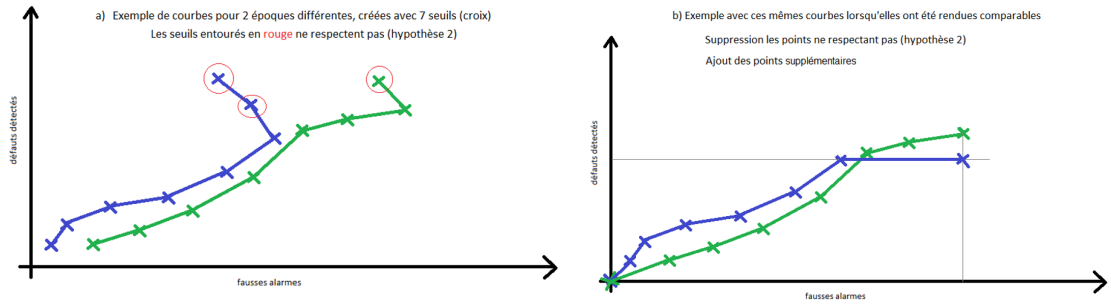

Figure B.5: Example of 2 curves corresponding to 2 epochs, created from 7 thresholds: a) curves are not comparable and some points do not respect (hypothesis 2); b) curves at the end of step 3

B.6.1.4 Final score

Let $M_{admitted}$ be the maximum number of false alarms that we accept to have on the whole validation base in a sub-optimal operating regime (if $M_{admitted} > M$, then we fix $M_{admitted} = M$), for example $M_{admitted}$ can be equal to the number of images in the validation set. We then consider all the curves for abscissas between 0 and $M_{admitted}$. For each curve, if there is no point S with an abscissa equal to $M_{admitted}$, we find the coordinates of such a point by linear interpolation. We then calculate the area under the curve between 0 and $M_{admitted}$, which gives us the final score. The higher the score, the better. Thus, for the ensemble methods, we use the epochs for which this score is the largest.

B.6.2 Test

During the testing process (see FigureB.6), the parts to inspect are tested only for the epochs selected during calibration and ensemble methods are then used to merge the results.

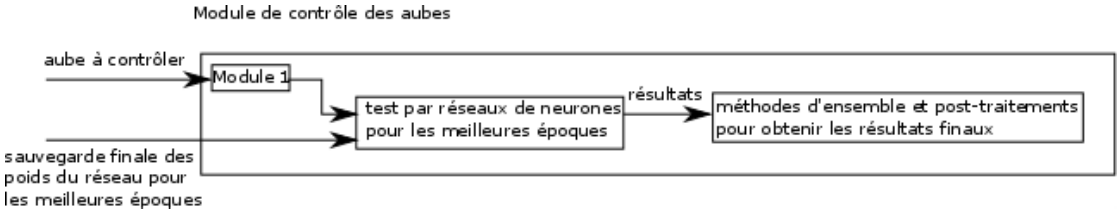

Figure B.6: Scheme of operation in test

- *Modifications for a classification network*

It is possible to apply this method with the use of a classification neural network. In this case, instead of returning a matrix of scores for each pixel of the patch, the network returns a single score, determining whether the patch contains a defect or not. For the annotation of the patch, we also have a single score: classically 1 if the patch contains a defect, 0 otherwise.
In this case we modify step 1 by creating a matrix M_{patch} of the same size as the patch and by putting the score returned by the network at all the locations in the matrix. The rest is unchanged.

B.7. Usage recommendations or limitations

In order for the presented method to perform better than a standard multi-epoch consolidation method, a large part of the anomalies must be large enough to span several patches. It is therefore recommended that the size of the patches and their overlap be adapted to meet this condition. However, it is recommended that patches are not made too small, as this may result in a loss of texture context.
Limitations: this method is created to improve results of a neural network, which takes patches as input. If this neural network performs bad, it is useless to try to improve it with this method.

C. Normalizing Flows

In this section, we present a OOD detection method using Normalizing Flows (NF) developed by the DEEL program. This architecture, called Hamiltonian Normalizing Flow (HNF), is inspired from a Monte Carlo Markov Chain (MCMC) algorithm, i.e. Generalized Hamiltonian Monte Carlo (GHMC). All the details of this method are in Chapdeleine (2021).

C.1. Objective

To detect OOD data, several strategies can be applied. In particular, one may learn a parametric generative model of the training ID data and compute for each new data score for being OOD or not, based on this generative model. Among generative models, NF have the interesting property to be invertible and to compute an estimation of the log-likelihood for each data point. As the NF is learn only with ID data, the log-likehood of OOD data should be worse, which allows, thanks to a threshold, to distinguish ID and OOD data during test time.

C.2. Inputs and outputs

This method requires as input :

- A dataset of ID images for the training.
- A dataset of ID and OOD images for the testing.

This method produces as output :

- Log-likehood of the monitored image.

C.3. Monitoring class

This method is a generative model of the training ID data. During monitoring, a score is returned based on this generative model. Given a threshold, the monitored image is considered as ID or OOD.

C.4. Design principles

We present in this section the design principles citing Chapdeleine (2021). This model is inspired from a Monte Carlo Markov Chain (MCMC) algorithm called Hamiltonian Monte Carlo (HMC) Neal (2012). Starting with a random sample from a simple distribution (such as standard Gaussian), MCMC algorithms propose strategies to efficiently update the sample, so it converges to a sample of a target distribution, which is known up to a normalizing constant. Among all MCMC algorithms, HMC has the property that, up to its Metropolis-Hastings steps, its update equations are reversible. Interestingly, one may notice that NF try to do the reverse way of MCMC algorithms, mapping an input data to a feature point with tractable distribution (such as standard Gaussian). Taking advantage of this analogy, HNF uses the reversible update equations of HMC Neal (2012), and more particularly of its generalization with neural networks Levy et al. (2018), to map the input data to the feature space. As such, HNF is the first NF

architecture inspired from a sampling algorithm, while previously proposed NF architectures, such as Real NVP Dinh et al. (2016) and Glow Kingma and Dhariwal (2018), are built in a more exploratory way, trying to find reversible operations with tricks to efficiently compute the log-determinant of the Jacobian.

C.5. Detailed design

Detailed design is provided in Chapdeleine (2021). We cite here some interesting points as a summary.
This algorithm is inspired from MCMC algorithms, to propose a new type of NF, i.e. HNF.. Indeed, an analogy can be made between MCMC algorithms and NF. While MCMC algorithms start from samples from a simple distribution and transform it to get samples from the real life, NF apply the reverse way, starting from samples from the real life to transform it into samples from a simple distribution. As an advantage with respect to other MCMC, HMC and GHMC have reversible procedures to propose samples. In addition, GHMC benefits from a fast convergence which makes it able to return samples of good quality in few time steps. Therefore, HNF is derived from GHMC. To preserve the invertibility, the Metropolis-Hastings steps are not included in the flow.
As in other NF architectures, the features in each layer of HNF are split into two parts in order to easily ensure invertibility. As in HMC, a separable Hamiltonian is defined: $H(z,a) = U(z) + V(a)$. The energies $U(z)$ and $V(a)$ have to be learnt for the OOD detection. The update of the position z and the momemtum a are detailed in Chapdeleine (2021). These updates allows to introduce the full leapfrog coupling layer in HNF, which computes the next features (z_{t+1}, a_{t+1}) from the previous features (z_t, a_t). As Real NVP Dinh et al. (2016), HNF uses a multiscale architecture. Since in one leapfrog coupling layer, the features with the role of the momentum are more updated than the features with the role of the position, the roles of the features are reversed between two successive leapfrog coupling layer, so interactions between all the dimensions are promoted.Concerning the splitting $s = (z,a)$ of the features in each scale, Real NVP Dinh et al. (2016) performs it spatially then channelwise after a squeezing operation. Differently, Glow Kingma and Dhariwal (2018) only performs channelwise splitting to simplify the architecture. The authors of HNF do the same choice as Glow and only apply channelwise splitting in HNF in order to alleviate the memory cost of the model. Given these updates, Chapdeleine (2021) presents the learning of the energies. Since the energies are unknown in the setting of OOD detection, the gradients of the energies are represented by neural networks. These gradient networks are learnt in the same time as the scaling and translation networks. Notably, in one scale of HNF, the gradient networks are the same used in all the leapfrog coupling layers of the scale. We don't detail here the formulae of these networks, but schematically the gradient networks look like autoencoders, with bridges between each layer of the encoder and the decoder.

C.6. Usage recommendations or limitations

Given the poor results on Safran Visual Industrial Control Use Case, while it had good results in the DEEL use case, we recommend to use this method in complement with another monitoring function or to deal with easy OOD.

D. Vision Transformers

D.1. State of the art

Transformer is an attention-based network first introduced in 2017 by Vaswani et al. (2017) and achieves brilliant success in natural language processing tasks. With the great success of transformer in that field, it was then adopted to the field of computer vision for the image classification tasks in 2020 with ViT Dosovitskiy et al. (2020). In this method, the image pixels are folded into word tokens and inputted to transformer. More generally, in vision transformers, convolution operations are replaced with attention-based transformer modules, to better capture long-range dependencies of image pixels. Given the good results with vision transformers, this method gained in popularity and a lot of state-of-the art results are achieved with transformer-based networks for various vision tasks.
Given the good results in various tasks, it has been decided to explore transformers for OOD and anomaly detection for Safran Visual Industrial Control use case. As vision transformers are quite recent, there were for the moment few attempts for our task of interest. Among the different methods, four recent methods (released in 2021 and 2022) caught our attention and given the availability of the code, we tempted to adapt three of them to the Safran Visual Industrial Control use case during batch 2. VT-ADL Mishra et al. (2021) apply transformer encoder to reconstruct images. Unfortunately during adaptation to the Safran Visual Industrial Control use case, the method was very unstable, which makes it incompatible with industry. Another approach, InTra Pirnay and Chai (2022) adopts transformer to recover the image by recovering all masked patches one by one. Unfortunately all available codes didn't achieve to reproduce the results of the paper and mail exchange with authors showed that InTra was done in a industrial context with confidential parts, so it seemed impossible to reproduce the good results with only information contained in the paper. More recently, AnoViT Lee and Kang (2022) is a method which apply transformer encoder to reconstruct images like VT-ADL, with a different architecture. This method was not tested as there was no available code and as the method didn't seem original enough compared to VT-ADL. Last method, UTRAD Chen et al. (2022) is also a reconstruction based method. However, it differs because reconstruction is done on a feature level via a pre-trained network. This method was tested and fully adapted to the Safran Visual Industrial Control use case given very interesting results during batch 2. During batch 3, this methods was also successfully adapted to the Renault Welding use case.
The conclusions from the results of UTRAD on Safran Visual Industrial Control and Renault Welding use cases are:

- UTRAD is well adapted to OOD of type Gaussian noise or superimposition (from Safran Visual Industrial Control use case).
- UTRAD is well adapted to OOD of type outside conditions variations, like point of view, background, illumination (from Renault Welding use case).
- UTRAD is well adapted to OOD of type anomaly only if there a no big variations of the seen object to inspect. It is the case for the Renault Welding use case, when taking full image (but not when dividing it into patches). However, the Safran Visual Industrial Control use case present a lot of different points of view and changes of texture, which makes UTRAD unsuitable.

Analysing all the methods found during batch 2, we notice that all of them are trained each time

only on one object (one point of view) of the public datasets. In other words, UTRAD (and other models) are trained only on object 1 to be deployed on object 1, only on object 2 to be deployed on object 2... When a model trained on all of the objects, in order to deploy one model whatever the object, the results during OOD detection are bad. This explains bad results on the Safran Visual Industrial Control use case, as each point of view of images and each localization of patches, could be considered as a different object. A unified model (one model which can work on different objects or different points of view) would be a preferred model to industrial use cases, however during batch 2 there no unified models in the state of the art.
During batch 3, we conducted a new state of the art. Three methods using Transformers caught our attention. AnoViT Lee and Kang (2022) encodes directly images instead to make reconstructions of features extracted by a backbone, like in other methods. HaloAE Mathian et al. (2022) uses data augmentation included in the architecture. However, like UTRAD, these models are not unified models. The only Transformers-based unified model we found during this state of the art is UniAD You et al. (2022). In this model, three improvements assist the model against learning identical shortcut, where both normal and anomalous samples can be well recovered, and hence fail to spot outliers. This method was tested and partially adapted to the Safran Visual Industrial Control and Renault Welding use cases. Results, which are presented in the Anomaly Detection Models Use Case level document, are however not as good as expected, as they show no real improvement in compa
A very complete survey on industrialisation anomaly detection Liu et al. (2023) (January 2023) presents all existing type of methods (not only Transformer based). The authors notice that it is challenging to enable the creation of a unified industrial anomaly detection (IAD) model in the absence of multiple domain IAD datasets. Like in our study, the only IAD model they found in the state of the art is UniAD. However, Liu et al. (2023) notice that the authors of UniAD disregard the notion that commodities produced in the same plant should be of the same sort. For example, an automaker manufactures several types of workpieces but does not produce fruit. Indeed, current popular IAD datasets, like MVTec AD Bergmann et al. (2019) and MVTec LOCO Bergmann et al. (2022), consist of numerous classes but not multiple domains. Consequently, to simulate a realistic manufacturing process, the community of IAD should create a new IAD dataset collected from multiple domains.
During batch 3, we found for a first time a unified anomaly detection method Lu et al. (2023) (October 2023, ICCV) adapted to industrial public datasets using Diffusion Models, an emerging field for deep computer vision. Consequently, this method should be considered for further exploration of new OOD reconstruction-based methods.

D.2. Objective

To detect OOD data, several strategies can be applied. In particular, in reconstruction-based methods, a generating model is optimized on a training dataset with only ID samples, thus learning good representations of ID samples. During inference, the generating model is used to reconstruct the input images, while the not-well-constructed images or image regions can be identified as anomalies or OOD. We successfully adapted such a reconstruction and transformers-based method UTRAD Chen et al. (2022) and UniAD You et al. (2022).

D.3. Inputs and outputs

The studied methods UTRAD and UniAD require as input :

- A dataset of ID images for the training.
- A dataset of ID and OOD images for the testing.

The studied methods UTRAD and UniAD produce as output :

- A score for each image

D.4. Monitoring class

The studied methods UTRAD and UniAD are reconstruction models of the training ID data. During monitoring, a score is returned based on the achieved reconstruction. Given a threshold, the monitored image is considered as ID or OOD.

D.5. Focus on UTRAD

D.5.1 Design principles

The overall framework of the UTRAD approach (Chen et al. (2022)) is shown in FigureD.1. First the method extracts multi-scale features for the input sample using a pre-trained CNN backbone. Then the proposed U-Transformer is used as a multi-scale reconstruction model for feature reconstruction. The feature reconstruction tends to fail with OOD samples, since the U-Transformer is trained with only ID samples. Therefore, the reconstruction error is adopted as a metric of OOD scores.

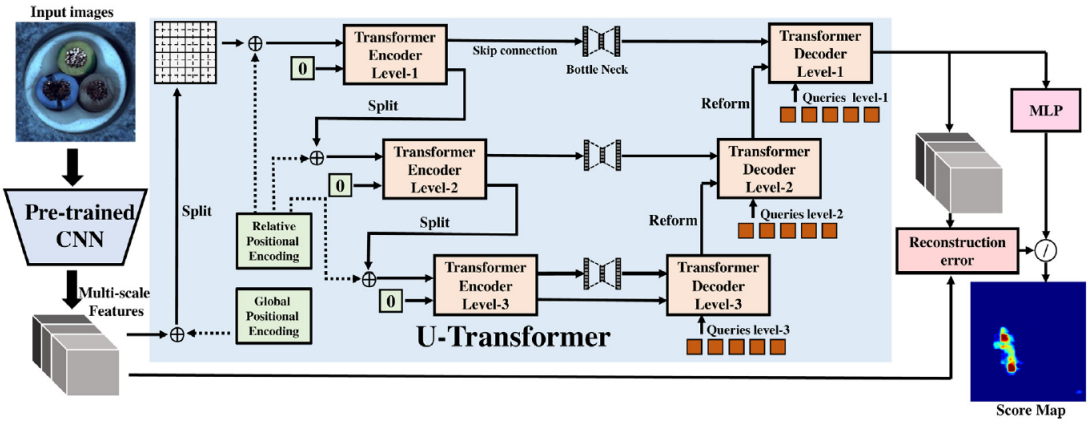

Figure D.1: Overall framework of the UTRAD method, from Chen et al. (2022)

D.5.2 Detailed design

In this section we summarize the detailed design of the UTRAD method. Even more details are in Chen et al. (2022). As it has been proven that normal samples and anomaly samples are much more distinguishable in feature space, UTRAD uses a frozen pre-trained CNN backbone to extract multi-scale features. This backbone is ResNet-18 pre-trained on ImageNet. It is composed of 4 layers and the UTRAD method use the mid-layer feature maps from layer 1 to layer 3. The 3 feature maps are upsampled to the same size firstly, then concatenated along the channel axis to form the multi-scale feature map. The obtained deep feature map differs from the 2-D image data. Indeed, in the feature map, each feature vector corresponding to the image pixel

can be processed independently with its surrounding vectors. Besides, the feature vector still contains regional information and local information related to the original pixel. Therefore, such feature embeddings are quite suitable to be handled as "word" tokens, under a transformer-based network, discretely and separately. A transformer-based reconstruction network U-Transformer is proposed to perform feature-level reconstruction. This U-Transformer takes these feature embeddings as input and outputs their reconstruction. Mean-Squared Error (MSE) loss is chosen as reconstruction loss to train this network. The reconstruction error is used to generate the anomaly score map. However, the reconstruction error varies on normal features. Therefore UTRAD adopts an MLP network to estimate the variance. The output of the MLP network is used as a scale factor to refine the reconstruction error map to the score map. We can notice that the original transformer affords too high calculation complexity, however UTRAD provides a U-Transformer, a multi-level transformer with skip connections, with much lower calculation complexity. This transformer is described in four parts:

1. Multi-level transformer encoder. Each feature map is divided into patches, which are tokenized with an additional zero padding token to serve as input tokens of the level-1 transformer encoder. The outputs of this transformer encoder are also tokens including feature latent vectors and a header latent vector. The feature latent vectors serve as parts of the inputs of the same level of transformer decoder, which is described in Multi-Level Transformer Decoder. The header latent vector serves as an overall feature embedding of the whole patch, which is used as the input of the level-2 transformer encoder. The header latent vectors are reshaped as a feature map. Then the same processes including patch split, tokenization, and padding are adopted to get the input tokens of the level-2 transformer encoder. It is similar to construct a level-3 transformer encoder from a level-2 transformer encoder. In practice, the level number of transformer encoder is set as 3.
2. Skip connections. The skip connection component is designed to combine feature embeddings from different levels and keep low-level details for a more precise anomaly localization result. For the outputs of transformer encoders, the feature latent vectors are used as parts of inputs of the transformer decoder at the same level. A bottleneck is added to each skip connection to restrain the low level skip connections and diffuse the information flow.
3. Multi-Level Transformer Decoder. Each level of transformer decoder takes three parts of inputs including the residual embeddings from skip connection, the output of the higher level transformer decoder, and the query embeddings. The query embeddings are learnable embeddings, which have the same shape with the corresponding residual embeddings. Also, the outputs of the transformer decoder share the same shape with the query embeddings.
4. Score map multi-level smoothing. The feature embeddings are reconstructed level-by-level, which means that the score of a feature embedding could be influenced by other feature embeddings in the same patch. Therefore, the score map is discrete with obvious discontinuity between different patches. To weaken the impact of this problem, UTRAD decomposes the score map into 3 score maps, which correspond to the 3 levels of U-Transformer, respectively. The 3 score maps are averagely smoothed with kernel sizes equal to the level scales, and then added together to form the final score map.

D.5.3 Usage recommendations or limitations

The conclusions from the results of UTRAD on Safran Visual Industrial Control and Renault Welding use cases are:

- UTRAD is well adapted to OOD of type Gaussian noise or superimposition (from Safran Visual Industrial Control use case).
- UTRAD is well adapted to OOD of type outside conditions variations, like point of view, background, illumination (from Renault Welding use case).
- UTRAD is well adapted to OOD of type anomaly only if there a no big variations of the seen object to inspect. It is the case for the Renault Welding use case, when taking full image (but not when dividing it into patches). However, the Safran Visual Industrial Control use case present a lot of different points of view and changes of texture, which makes UTRAD unsuitable.

For the last point (anomaly detection), UTRAD is adapted when trained and deployed on the same object (for example one type of welding) and on the same domain (for example same illumination, texture). If the object to inspect present multiple domains or if one model has to be deployed on multiple objects, UTRAD is no more suitable for anomaly detection. A unified model should be created or adapted.

D.6. Focus on UniAD

D.6.1 Design principles

The overall framework of the UniAD approach (You et al. (2022)) is shown in FigureD.2. The framework consisting of a Neighbor Masked Encoder (NME) and a Layer-wise Query Decoder (LQD). First, the feature tokens are extracted by a fixed pre-trained backbone and integrated by NME to derive the encoder embeddings. Then, in each layer of LQD, a learnable query embedding is successively fused with the encoder embeddings and the outputs of the previous layer (self-fusion for the first layer). The feature fusion is completed by the Neighbor Masked Attention (NMA). The final outputs of LQD are viewed as the reconstructed features. Also, there is a Feature Jittering strategy to add perturbations to the input features, leading the model to learn normal distribution from the denoising task. Finally, the results of OOD localization and detection are obtained through the reconstruction differences.

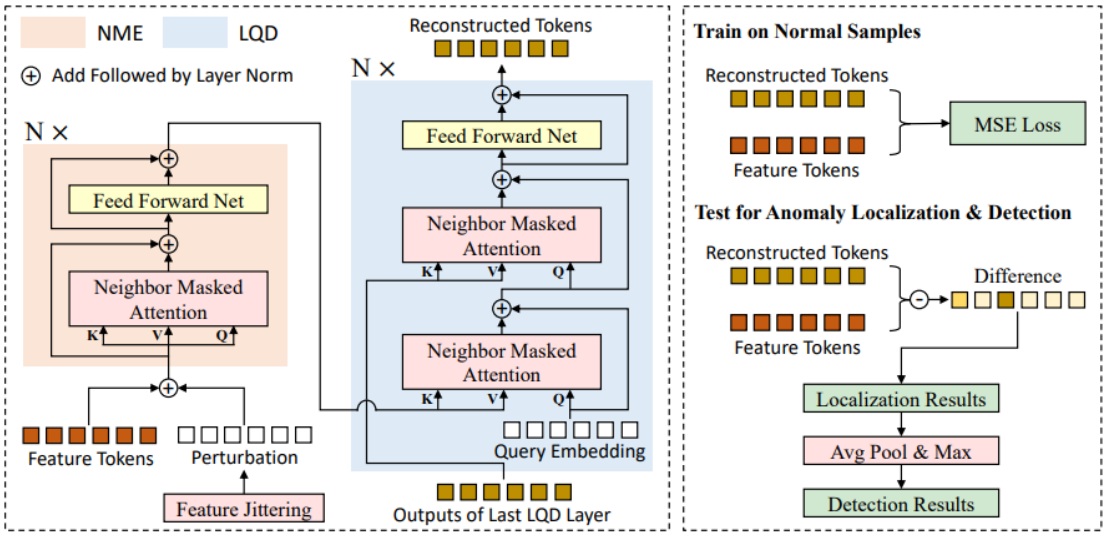

Figure D.2: Overall framework of the UniAD method, from You et al. (2022), consisting of a Neighbor Masked Encoder (NME) and a Layer-wise Query Decoder (LQD).

D.6.2 Detailed design

In this section we summarize the detailed design of the UniAD method. Even more details are in You et al. (2022). Like in UTRAD, UniAD uses a frozen pre-trained CNN backbone to extract multi-scale features. Experiments has been done with several architectures of ResNet and EfficientNet pre-trained on ImageNet, using the first 3 or 4 layers of these models. Like in UTRAD, all the 3 or 4 feature maps are resized to the same size, then concatenated along the channel axis to form the multi-scale feature map. This feature map is first tokenized, followed by a linear projection to reduce channel dimension. Then these tokens are processed by NME and LQD (more details following). The learnable position embeddings are added in attention modules to give the spatial information. Afterward, another linear projection is used to recover the initial channel dimension. After reshape, the reconstructed feature map, is finally obtained. Mean-Squared Error (MSE) loss is chosen as reconstruction loss to train this network. Like in UTRAD, the reconstruction error is used to generate the anomaly score map. In order to prevent the identical shortcut, the authors makes several changes to classif Transformer architecture:

- Neighbor masked attention (NMA). The authors suspect that the full attention contributes to the identical shortcut as one token is permitted to see itself and thus is easy to reconstruct by simply copying. Moreover, considering that the feature tokens are extracted by a CNN backbone, the neighbor tokens must share lots of similarities. Therefore, UniAD proposes to mask the neighbor tokens when calculating the attention map.
- Neighbor masked encoder. The encoder of UniAD follows the standard architecture of vanilla transformer, however the full attention is replaced by the proposed NMA to prevent the information leak
- Layer-wise query decoder (LQD). The authors showed that the query embedding of the vanilla transformer is of vital significance to model the normal distribution. However, there is only one query embedding in the vanilla transformer. Therefore, UniAD adds the query embedding in each decoder layer. Specifically, in each layer of LQD, a learnable query embedding is first fused with the encoder embeddings, then integrated with the outputs of the previous layer (self-integration for the first layer). The feature fusion is implemented by NMA. Following the vanilla transformer, a 2-layer FFN is applied to handle these fused tokens, and the residual connection is utilized to facilitate the training. The final outputs of LQD serve as the reconstructed features.
- Feature jittering. Feature Jittering (FJ) strategy is employed to disturb the input features, adding perturbations from a Gaussian distribution to feature tokens, leading the model to learn normal distribution from denoising.

D.6.3 Usage recommendations or limitations

As conclusion, given the results on the Safran Visual Industrial Control and Renault Welding use cases, UniAD doesn't seem very different from UTRAD, consequently the usage recommendations and limitations are the same than for UTRAD.

E. Diffusion Models

E.1. State of the art

Recently, diffusion models have gained popularity as prolific deep generative models. Diffusion models, a class of generative models inspired by non-equilibrium thermodynamics, define a paradigm in which the forward process slowly adds random noise to the data, and the reverse constructs the desired data samples from the noise Zhang et al. (2023a). Recently, a wide range of diffusion-based perception applications has emerged, such as image generation, image segmentation, object detection, etc. Several 2022 and 2023 methods explores the application of diffusion models to reconstruct anomalies on industrial or medical public datasets. We present them chronologically.

AnoDDPM Wyatt et al. (2022) is the first approach to employ a diffusion model for medical anomaly detection. However, it has high false positive rate. In addition, its iterative denoising approach leads to a notably slow inference speed and high computational cost.

In 2022, Teng et al. (2022) introduce score based generative model into the unsupervised defect detection task. The anomalous samples, injected with noise, are projected into normal data distribution space through two separate SDE functions. They obtain anomaly masks by separately extracting feature maps and calculating distances. However, they are unaware of the large sample variation in the same class, resulting in unpleasant performance.

Two papers were presented at ICCV 2023. In Lu et al. (2023), the authors proposed to simultaneously predict the noise and generate features that mimic the features extracted from a pretrained convolutional neural network. Their model is robust to various anomaly types and can be extended as an unified anomaly detector for all categories like UniAD You et al. (2022). DiffAD Zhang et al. (2023b) proposes using latent diffusion models instead of traditional autoencoder-based sub-networks for reconstruction. It also introduces noisy condition embedding and interpolated channels to guide the generation and reduce misalignment between the reconstructed and original images. DiffAD has better results on MVTec-AD than Lu et al. (2023) in baseline usage, but there is no results on the unified case.

Several papers were released on ArXiv in November and December 2023. TransFusion Fučka et al. (2023) proposes a method where the transparency of the anomalous regions is gradually increased, effectively removing them, and restoring their normal appearance. DiffusionAD Zhang et al. (2023a) utilizes an anomaly synthetic strategy to generate anomalous samples and labels, along with two sub-networks dedicated to the tasks of denoising and segmentation. Firstly, it redefines the reconstruction process using a diffusion model. Secondly, it presents a one-step denoising paradigm, which is significantly faster than the iterative denoising approach in diffusion models. In addition, it proposes the norm-guided paradigm to further enhance the reconstruction quality. DDAD Mousakhan et al. (2023) employs a score-based pre-trained diffusion model to generate normal samples while finetuning the pre-trained feature extractor to achieve domain transfer. DiAD He et al. (2023) addresses the issue of category and semantic loss in the stable diffusion model for multi-class anomaly detection. It proposes a Semantic-Guided network and a Spatial-aware Feature Fusion block to better reconstruct the abnormal regions while maintaining the same semantic information as the input image. While the four methods present impressive results, only DiAD has been tested on a multi-class setting. These four methods has to be considerate carefully in next research as they are very recent and not yet peer-reviewed.

E.2. Conclusion on diffusion models

After a slow beginning in 2022, diffusion models for anomaly detection by reconstruction became very popular in 2023. They produce very impressive results and some methods propose a unified or multi-class setting. Theses methods should be considerate for further research on industrial use case of Confiance.ai program.

F. Conclusion

In this document, we presented different OOD and anomaly detection methods.
First, we presented a new design method for improving the detection of OOD (classification or semantic segmentation). This method can be applied to both non-OOD and OOD anomaly detection. This method can be easily extended to other type of OOD detection. It improves detection as long as a neural network is used to solve this task, and as long as patches of images (instead of full images) have to be the input of the neural network. This method uses ensemble methods with fusion of results from a neural network taken at different convergence points. The added value is how to select the convergence points.
Next, we presented an OOD detection method using Normalizing Flows (NF) developed by the DEEL program. This method can separate OOD and ID data for only if there a no big variations of the seen object to inspect or if the OOD and ID data is very different.
Then, we presented the OOD detection by reconstruction methods with Transformers. Two methods from state of the art have been successfully tested on Safran Visual Industrial Control and Renault Welding use cases. We presented these methods and how to use them. These methods should be used for detection of OOD of type added noise or superimposition, variation of outside conditions. They can also detect anomalies but only if there a no big variations of the seen object to inspect.
Last, we presented a brief state of the art study for OOD detection with diffusion models. After a slow beginning in 2022, diffusion models for anomaly detection by reconstruction became very popular in 2023. They produce very impressive results and some methods propose a unified or multi-class setting.
As perspectives, it would be interesting to considerate in more details the methods using diffusion models, adapting one or several best methods from the state of the art, to the industrial use cases of the Confiance.ai program.

Bibliography

Bergmann, P., Batzner, K., Fauser, M., Sattlegger, D., and Steger, C. (2022). Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization. *International Journal of Computer Vision*, 130.

Bergmann, P., Fauser, M., Sattlegger, D., and Steger, C. (2019). Mvtec ad — a comprehensive real-world dataset for unsupervised anomaly detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9584–9592.

Chapdeleine, C. (2021). Out-of-distribution (ood) detection with hamiltonian normalizing flows (hnf). Technical Report V075L02T00-007, DEEL.

Chapdeleine, C. and Picard, S. (2020). Procédé de contrôle d'une pièce mécanique par apprentissage en ligne de réseaux de neurones et dispositif associé.

Chen, H., Lundberg, S., and Lee, S.-I. (2017). Checkpoint ensembles: Ensemble methods from a single training process.

Chen, L., You, Z., Zhang, N., Xi, J., and Le, X. (2022). Utrad: Anomaly detection and localization with u-transformer. *Neural networks : the official journal of the International Neural Network Society*, 147:53–62.

Cheng, J., Aurélie, n. B., and Mark, v. d. L. (2018). The relative performance of ensemble methods with deep convolutional neural networks for image classification. *Journal of Applied Statistics*, 45(15):2800–2818.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2016). Density estimation using real nvp. cite arxiv:1605.08803Comment: 10 pages of main content, 3 pages of bibliography, 18 pages of appendix. Accepted at ICLR 2017.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale.

Fučka, M., Zavrtanik, V., and Skočaj, D. (2023). Transfusion – a transparency-based diffusion model for anomaly detection.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 1050–1059.

He, H., Zhang, J., Chen, H., Chen, X., Li, Z., Chen, X., Wang, Y., Wang, C., and Xie, L. (2023). Diad: A diffusion-based framework for multi-class anomaly detection.

Kingma, D. P. and Dhariwal, P. (2018). Glow: Generative flow with invertible 1x1 convolutions. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*.

Lee, Y. and Kang, P. (2022). Anovit: Unsupervised anomaly detection and localization with vision transformer-based encoder-decoder. *IEEE Access*, 10:46717–46724.

Levy, D., Sohl-dickstein, J., and Hoffman, M. (2018). Generalizing hamiltonian monte carlo with neural networks.

Liu, J., Xie, G., Wang, J., Li, S., Wang, C., Zheng, F., and Jin, Y. (2023). Deep industrial image anomaly detection: A survey.

Lu, F., Yao, X., Fu, C.-W., and Jia, J. (2023). Removing anomalies as noises for industrial defect localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16166–16175.

Mathian, E., Liu, H., Fernandez-Cuesta, L., Samaras, D., Foll, M., and Chen, L. (2022). Haloae: An halonet based local transformer auto-encoder for anomaly detection and localization.

Mishra, P., Verk, R., Fornasier, D., Piciarelli, C., and Foresti, G. L. (2021). Vt-adl: A vision transformer network for image anomaly detection and localization. In *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, pages 01–06.

Morard, V. (2018). Procédé de contrôle non-destructif de pièce métallique.

Morard, V. and Parra, E. (2017). Procédé de contrôle non-destructif par apprentissage.

Mousakhan, A., Brox, T., and Tayyub, J. (2023). Anomaly detection with conditioned denoising diffusion models.

Neal, R. M. (2012). Mcmc using hamiltonian dynamics. cite arxiv:1206.1901.

Pirnay, J. and Chai, K. (2022). Inpainting transformer for anomaly detection. In Sclaroff, S., Distante, C., Leo, M., Farinella, G. M., and Tombari, F., editors, *Image Analysis and Processing – ICIAP 2022*, pages 394–406, Cham. Springer International Publishing.

Teng, Y., Li, H., Cai, F., Shao, M., and Xia, S. (2022). Unsupervised visual defect detection with score-based generative model.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wyatt, J., Leach, A., Schmon, S. M., and Willcocks, C. G. (2022). Anoddpm: Anomaly detection with denoising diffusion probabilistic models using simplex noise. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 649–655.

You, Z., Cui, L., Shen, Y., Yang, K., Lu, X., Zheng, Y., and Le, X. (2022). A unified model for multi-class anomaly detection. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 4571–4584. Curran Associates, Inc.

Zhang, H., Wang, Z., Wu, Z., and Jiang, Y.-G. (2023a). Diffusionad: Norm-guided one-step denoising diffusion for anomaly detection. *arXiv preprint arXiv:2303.08730*.

Zhang, X., Li, N., Li, J., Dai, T., Jiang, Y., and Xia, S.-T. (2023b). Unsupervised surface anomaly detection with diffusion probabilistic model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6782–6791.

Confiance ai

Title: Methodological Guideline for Anomaly Detection Models

Keywords: Data driven out-of-distribution detection, neural networks

This document presents several out-of-distribution data-driven detection methods, based on neural networks, particularly adapted to the Safran Visual Industrial Control and Renault Welding use cases. Among these methods, there is a new design method for improving the detection of out-of-distribution data by a neural network using ensemble methods, and two reconstruction based methods using Normalizing Flows or Vision Transformers. One reconstruction based method with Normalizing Flows from the DEEL project is also presented. This document contains also a small state on the art on reconstruction based method by Diffusion Models.

Our partners:

Air Liquide, AIRBUS, Atos, cea, Inria, NAVAL GROUP, GROUPE RENAULT, SAINT EXUPÉRY, SAFRAN, sopra steria, SystemX, THALES Building a future we can all trust, Valeo, Investir l'Avenir