University of Padova

Department of Physics and Astronomy "Galileo Galilei"

Master Thesis in Physics of Data

Out-of-distribution detection methods on deep neural network encodings of images and tabular data

Supervisor
Prof. Marco Baiesi
University of Padova

Internal advisors
Jacopo Credi
Giovanni Davoli
Axyon AI Srl

Master Candidate
Simone Mistrali

Student ID
2014119

Academic Year
2021-2022

"I have approximate answers, and possible beliefs, and different degrees of certainty about different things, but I'm not absolutely sure of anything."

— Richard P. Feynman.

Abstract

Anomaly detection is the process of finding patterns in data that do not conform to a model of normal behaviour.

The goal of anomaly detection is to identify cases that are unusual within data that is seemingly comparable. Anomaly detection is an important tool for detecting fraud, network intrusion or even new physics, *i.e.* rare events that may have great significance but are hard to find.

Anomaly detection can be used effectively as a tool for risk mitigation and fraud detection.

Today, data drives most business and other important decisions in our everyday life. With access to more data and more information than ever before, it is even more important to analyze it and interpret it correctly, for example when it comes to security, finding the outliers is only the first step. Determining if the outlier is a security threat, and understanding the root cause of the anomaly is the key to a real solution.

The scope of this thesis work is to implement an anomaly detector pipeline, capable of detecting the anomalous and normal samples in the test set of a deep learning classifier. The pipeline is divided into two steps: the preprocessing part, the idea, in this case, is to use the compact representation given by the inner layer of a trained neural network on a ”clean” dataset, and an anomaly detector, which scope is to recognize if the sample is an anomaly. The work was initially developed during the internship experience in the start-up Axyon AI. The application of the algorithm in this context is the recognition of economic anomalies. The idea is to use the pipeline created paired with a classifier to give an insight into the ”quality” of the data where the classifier makes the prediction.

The thesis is organized into five chapters. The first one explores what is novelty detection and which are the most used anomaly detectors, using both machine learning and classical techniques. An extensive analysis of the detectors used in this work is detailed in chapter two. Use cases in both physics and the business world are inspected in chapter three. Chapter four puts attention to the choice of the so-called toy dataset to train the algorithms. The details of the three experiments done in this work are given in chapter five, where the results and the performance on both the toy and real dataset are shown. As is underlined in chapter five the idea to use the embedding of a neural network to improve the score of the anomaly detection algorithms is promising, so the forward steps are to increase the complexity of the dataset and to use more powerful anomaly detectors.

Contents

Listing of figures

Listing of acronyms

ANN ..... Artificial Neural Network.

NN ..... Neural Network.

AD ..... Anomaly detection.

MLP ..... Multilayer perceptron.

CNN ..... Convolutional neural network.

FNN ..... Feedfoward neural network.

ML ..... Machine Learning.

t-SNE ..... t-distributed stochastic neighbor embedding.

IoT ..... Internet of things.

SVM ..... Support Vector Machines.

***k*-NN** ..... k-nearest neighbors.

ECOD ..... Empirical Cumulative distribution for Outlier Detection.

ECDF ..... Empirical Cumulative distribution function.

CDF ..... Cumulative distribution function.

LHC ..... Large Hadron Collider.

PCA ..... Principal component analysis.

MNIST ..... Modified National Institute of Standards and Technology.

OAL ..... One Anomalies Label.

RAL ..... Random Anomalies Labels.

ROC ..... eceiver Operating Characteristic.

AUC ..... Area under the ROC Curve.

AVPR ..... Average Precision score.

1 Introduction

Novelty detection is the identification of new data that machine learning systems, from now on the classifiers, are not aware of during the training time. In a nutshell, the novelty detection methods try to identify the outliers that differ from the distribution of ordinary data. Novelty, also called anomaly or outlier, is a pattern in the data that does not conform to the expected behaviour [4]. Novel events occur relatively infrequently so in general the data is insufficient to construct explicit models for "non-normal" classes, *i.e.* create a novel class[4].

1.1 Design a novelty detection pipeline

A general idea of a novelty detection pipeline is presented in Fig.1.1, in the case of this work the idea behind the Preprocessing part was to use an inner layer of a neural network, trained only on non-anomalous data, as a "pre-processor". The obtained samples are passed through the novelty detection algorithm to extract the probability that the sample is anomalous.

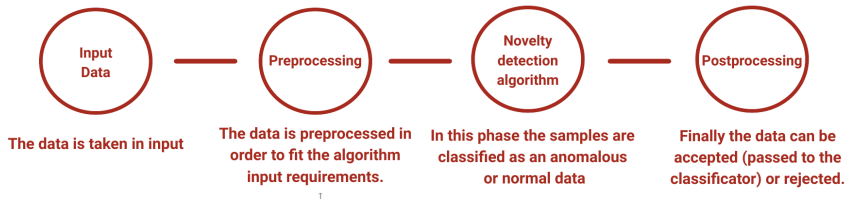


Figure 1.1: Novelty detection pipeline.

The idea behind the anomaly detectors is that exists a so-called "normal" region, a region in the feature space where the non-anomalous data can be easily clustered as can be seen in Fig. 1.2.

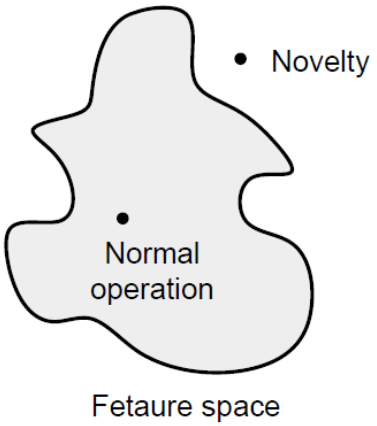


Figure 1.2: Novelty within feature space.

1.2 Mathematical definition of anomalies

Conventional pattern recognition typically focuses on the classification of two or more classes. For example in a two-class classification problem a set of training examples is given $\mathbf{X}_{train} = \{(\mathbf{x}_i, y_i | \mathbf{X}_i \in \mathbb{R}^D,\ i = 1, \ldots, N)\}$ where each sample consists of a D dimensional vector $\mathbf{x}_i$ and its label $y_i \in \{1, -1\}$. From the labelled dataset, a function, *i.e.* the classifier, $h(\mathbf{x})$ is constructed such that for given input vector $\mathbf{x}^*$ an estimate of one of the two labels is obtained:

$$\hat{y}^* = h(\mathbf{x}^* | h(\mathbf{X}))$$
$$h(\mathbf{x}^* | h(\mathbf{X})) : \mathbb{R}^D \rightarrow [-1; 1]$$

Novelty detection, most of the time, is approached within the framework of the one-class classification, in which one class (the normal one) has to be distinguished from all other possibilities. The problem, in this case, is that the positive class is in general very well sampled while the others are markedly under-sampled. The scarcity of negative samples can be due to high measurement costs, the low frequency at which the abnormal event occurs or the emergence of new events, so in the majority of the cases is difficult if not impossible to obtain a very well-sampled negative class. In this framework, normal patterns $\mathbf{X}_{normal}$ are available for training while abnormal ones are relatively few. In this case a model of normality $M(\vartheta)$, where ϑ represents the free parameters of the model, is constructed and used to assign novelty scores $z(\mathbf{x}^*)$ to unseen test data $\mathbf{x}^*$. Larger novelty scores correspond to an increased abnormality concerning the model of normality. A novelty threshold k is defined such that $\mathbf{x}^*$ is classified normal if $z(\mathbf{x}^*) \leq k$, or abnormal otherwise. In this case $z(\mathbf{x}^*) = k$ set the boundaries of the normal region in the feature space, Fig.1.2.

1.3 The novelty detection as a high dimensionality problem

High dimensionality refers to datasets that have a large number of independent variables, components, features, or attributes within the data available for analysis. The complexity of the data analysis increases for the number of dimensions, requiring more sophisticated methods to process the data. Datasets are growing in terms of sample size n but even more in terms of dimension m. At the same time, in the era of big data, m is commonly misconstrued as high-dimensional, since thousands of dimensions are very common. In general, a data set can be called high dimensional when the number of dimensions m causes the effect of the "curse of dimensionality". It is common to have multiple dimensions in many application areas such as data mining and machine learning. The advent of cloud-based storage enables organizations to store massive amounts of detailed information easily. A financial organization may monitor its stock price every minute, every hour, or every day, and the stock price may be predicted with linear combinations of thousands of underlying traded portfolios, each of which is a dimension. The other sources might be science or biology-related: for example, healthcare data are known for having multiple dimensions, such as treatment plans, radiation therapy, and genetic background. The major difference between these examples (i.e., business and healthcare data) is the number of samples. The number of variables or components can be similar for each. However, the business data set can have millions of records, whereas the health data will rarely involve more than a few thousand samples[5]. A sophisticated approach handles the number of increasing dimensions without affecting accuracy in either situation. High-dimensional data is often referred to as multi-dimensional or multi-aspect or multi-modal data throughout the literature. A generalised term for multi-dimensional data structure, along with arithmetic operations viability can be referred to as a tensor and appears frequently in many web-related applications.

1.3.1 Curse of dimensionality

The term "curse of dimensionality" was first introduced by Bellman [6] to describe the problem caused by a rise in the number of dimensions or input variables. When the dimensionality of data increases, data size increases proportionally, leading to data sparsity, and sparse data is difficult to analyze. The curse of dimensionality affects anomaly detection techniques because the level of abnormal nature of the increasing dimensions can be obscured or even concealed by unnecessary attributes. Since outliers are defined as data instances in sparse regions, an inadequate discriminative region is observed in almost equally sparse locations of high-dimensional space.

The increase in dimensions makes data points scattered and isolated and makes it elusive to find the global optima of the dataset[5]. The more dimensions that are added to a dataset, the more complex it becomes, as each added dimension brings about a substantial number of false positives. The curse of dimensionality refers to several occurrences that emerge when analyzing and organizing data in high-dimensional spaces. The very nature of these occurrences is that, whenever there is an increase in dimensionality, the volume of the space increases proportionally, such that all other data points become sparse. This sparsity is challenging for any technique that requires a statistical value. Further, it produces numerous complications associated with other noise levels, which may be irrelevant features or unnecessary attributes, that could complicate or even conceal the data instances. This is the main reason why many algorithms struggle with high-dimensional data. As the number of dimensions

increases, statistical approaches such as distance measures become less useful, since the points become almost equidistant from each other, due to the curse of dimensionality.

High-dimensional data requires significant computational memory and brings a huge computational burden. For high-dimensional data, recognizing useful insights or patterns becomes complex and challenging. The simplest approach to handling high dimensionality problem is to minimize the features and can be better understood by studying either the intrinsic or embedding dimensionality of data sets. It is essential to understand the subtle difference between intrinsic and embedding dimensionality. Intrinsic dimensionality is the smallest variety of features that cover the full representation of the data; embedding dimensionality is the representation of the number of features or columns of the whole data space.

1.4 Methods of novelty detection

The methods of novelty detection can be separated mainly into three different branches, the so-called taxonomy is presented in Fig. 1.3.

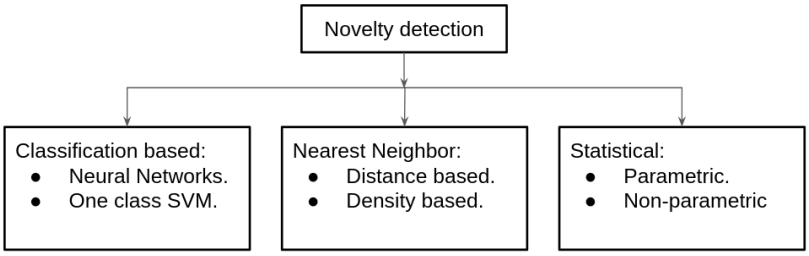


Figure 1.3: Taxonomy of novelty detection methods.

1.4.1 Classification based

In classification-based novelty detection classification model is built for normal (and anomalous but rare) patterns based on labelled training data, and such a model is used to classify each new unseen pattern.

Common classification-based approaches are:

- Neural Networks.
- One class SVM.

Neural Networks

The basic idea is to train the neural network on normal training data and then detect novelties by analyzing the response of the trained neural network to a test input. If the network accepts a test input, it is normal and if

the network rejects a test input, it is a novelty. This type of acceptance-rejection operation can be achieved using different approaches the most used in literature are: one class neural network, where the two classes are anomaly and normal data and reconstruction based where the samples are projected in low dimensional space and then "reconstructed" in the original space, after the training of this type of architecture the test set is projected and reconstructed and if a sample has a large reconstruction error is considered as an anomaly. Neural networks can generalize (the outputs of the network approximate target values given inputs that are not in the training set). The main neural networks architectures used in anomaly detection are:

- MLP: A multilayer perceptron is a feedforward artificial neural network model that maps sets of input data into a set of appropriate outputs. It is a modification of the standard linear perceptron in that it uses three or more layers of neurons (nodes) with nonlinear activation functions, and is more powerful than the perceptron in that it can distinguish data that is not linearly separable. The goal of the training process is to find the set of weight values that will cause the output from the neural network to match the actual target values as closely as possible. Training is accomplished by backpropagation algorithm [7].
- Autoencoder: In an autoencoder network each pattern presented to the network serves as both the input and the output pattern. Such networks consist of mapping (Encoder), middle (the latent space) and Decoder layer. The middle layer, with each node representing some feature of the environment, is used for compression or dimensionality reduction purposes, (a variant of PCA). During operation, patterns are presented to the network and compared to network outputs (ideally inputs and outputs should be the same). If the distance between the network's inputs and outputs is too large, the pattern is considered as novelty[8].

One class SVM

The SVM approach is based on the simple assumption that normal points belong to the high-density region and anomaly patterns to the low-density data region. The idea is to find a hyper-sphere to encompass all points in the data set with the minimum radius. Separation of non-spherical distributed data is done in high dimensional feature space into which vectors are mapped using non-linear mapping (kernel). The second approach with SVM assumes the existence of labelled data, in a semi-supervised manner, and uses standard SVM for classification.

1.4.2 Nearest neighbor

This approach is similar to the one-class SVM, it is based on the assumption that normal points have close neighbours while novelty points are located in the "low density" region. In this case, the most used algorithm is the k-NN algorithm where a point is classified by a "majority" vote of its neighbours. Two main variants of this approach are:

Distance based approach

In this variant, the point is considered a novelty if the distance to k-Near neighbour exceeds the predefined threshold.

Density based approach

This approach is preferable for a sparse dataset in this case the pattern which is further away is considered a novelty. This is achieved by computing the local outlier factor, a metric which gives an estimation of how strongly an instance can be considered a novelty.

1.4.3 Statistical approach

In most of the statistical approaches stochastic distribution is used to model the dataset, the two main approaches are:

Parametric approach

This approach assumes that normal data are generated from underlying parametric distribution, sch as a gaussian or any other statistical distribution. The limitation of parametric approaches for estimating the probability density function is that they require extensive prior knowledge of the problem.

Some examples of this approach are:

- Assume an underlying well-defined distribution: the most common distribution used is the Normal one, in this case, the pattern is considered a novelty when the probability density function falls below a threshold (often associated with 3σ distance from the mean μ).
- Extreme Value Theory: This approach investigates the distributions of data that have abnormally high or low values in the tails of the distribution that generates the data.
- Hypothesis testing: Novelty detection can be accomplished by determining whether the test samples come from the same distribution as training data or not using t-test. If t-test shows a significant difference between the two sets of measurements (normal profile and test profile), then the second set is considered to contain novel patterns.

Nonparametric approach

In this approach no assumption is made about the statistical distribution of the data. One simple example of this type of anomaly detection algorithm can be:

- Histogram: Histogram is a graphical display of tabulated frequencies, during normal operation of a system histogram acquired from collected data maintains a profile of normal data. The algorithm typically defines a distance measure between a new test instance and the histogram-based profile to determine if it is an outlier or not.

2 Models

Given the type of data used in the financial domain, the most interesting anomaly detectors seem to be:

- Isolation Forest.
- Autoencoders.
- Empirical Cumulative distribution for Outlier Detection (ECOD).

2.1 Isolation Forest

The *rationale* behind this type of model is to take advantage of two properties of the anomalies:

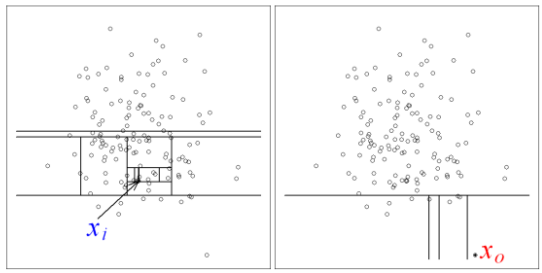
- By definition the anomalies are fewer than the normal data.
- the anomalies have attribute-values that are different from the normal instances.

So, in this case, the model pays attention on the anomalies, not the normal samples, meanwhile, in the other cases, the algorithms try to reconstruct the distribution of the data . Those differences can be exploited using a "Tree structure", using their susceptibility to isolation in the tree anomalies are isolated closer to the root of the tree, whereas the normal points are isolated deeper end to the tree [9]. This "Tree" is called Isolation Tree and one of the advantages of the tree based type of algorithm is that the only free parameters of this type of model are the number of trees to build and the sub-sampling size. Another good property of this algorithm is that works surprisingly good and converges quickly with a small number of trees.

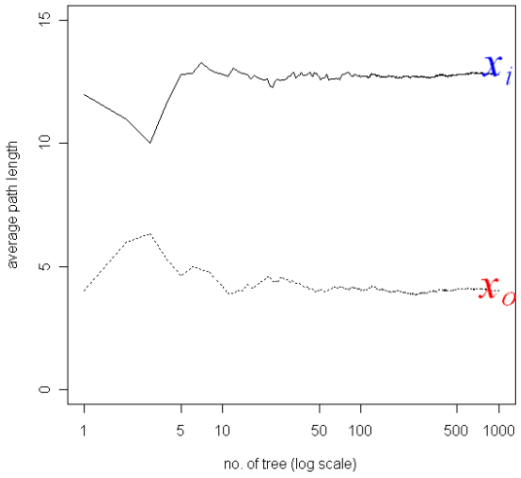
This algorithm is distinguished from the other model-based, distance based or density based models in the following ways:

- The isolation characteristic of the trees enables them to build partial models and exploit sub-sampling. Since a large part of a tree that isolates normal points is not needed for anomaly detection; it does not need to be constructed.
- Isolation Forest utilizes no distance or density measures to detect anomalies. This eliminates the major computational cost of distance calculation in all distance-based methods and density-based methods[1].

In a tree the data are partitioned recursively until all instances are isolated. The random partitioning produces shorter paths for anomalies since the fewer instances of anomalies result in a smaller number of partitions and shorter paths in a tree structure, and instances with distinguishable attribute values are more likely to be separated in early partitioning. So when a forest of random trees collectively produces shorter path lengths for some particular points, then they are highly likely to be anomalies as we can see in Fig.2.1.



(a) Isolating x_i (b) Isolating x_o



(c) Average path lengths converge

Figure 2.1: On top the plot of the samples, x_i is a normal point and x_0 is an anomaly, on the bottom the average path lengths of x_i and x_0 the plot is taken from [1].

To define the isolation forest algorithm some definitions are needed, an Isolation Tree is defined in the following way: let T be a node of an isolation tree. T is either an external node with no child, or an internal node with one test and exactly two daughter nodes (T_l, T_r). A test consists of an attribute q and a split value p such that the

test $q < p$ divides data points into T_l and T_r. Given a sample of data $\mathbf{X} = \{x_1, \ldots, x_n\}$ of n instances from a d-variate distribution, to build an isolation tree, $\mathbf{X}$ is recursively divide $\mathbf{X}$ by randomly selecting an attribute q and a split value p until the tree reaches a limit in its height or $\left|\mathbf{X}_{split}\right| = 1$ or all data in $\mathbf{X}$ have the same values. The idea behind anomaly detection is to provide a ranking which reflects "the degree of anomaly". One way to detect them is to sort data points according to their path lengths or anomaly scores. To define the anomaly score the path length has to be defined: $l(x^*)$ of a point x^* is measured by the number of edges x^* traverses in a Tree from the root node until the traversal is terminated at an external node. Defining the anomaly score is a key problem in the isolation forest algorithm because the single length of $l(x^*)$ grows linearly with the possible height of the Tree, but the average height grows in order of log n. The anomaly score s of an instance x is defined as:

$$s(x, n) = 2^{-\frac{\mathbb{E}(l(x))}{c(n)}} \tag{2.1}$$

where $c(n)$ is the average path length performed in a Binary Search Tree performed in a dataset of n samples: $c(n) = 2H(n-1) - 2\frac{2(n-1)}{n}$, $H(i)$ is the harmonic number $H(i) = \ln(i) + \varphi$, where φ is the Euler's constant, and $\mathbb{E}(l(x))$ is the average of $l(x)$ from a collection of isolation trees. During the training time the Tree is constructed by the partitioning of the given dataset until the instances are isolated or a given height is reached.[1]

2.2 ECOD: Empirical Cumulative distribution for Outlier Detection

ECOD is an outlier detector which tries to overcome two limitations of the most used anomaly detectors.
Firstly many of those detectors suffer the curse of dimensionality, secondly the majority of those methods require hyperparameter tuning. The idea behind ECOD is to exploit the definition of anomalous sample given the fact that the anomalies are rare, for example, if we assume a Gaussian underling distribution the three-sigma rule can be used. Using the three-sigma rule the histogram has to be constructed and some parameters such as the binning have to be tuned. ECOD estimates the empirical cumulative distribution function, so no parameters have to be tuned and no parametric assumption has to be done on the data. The problem in the estimation of the ECDFs is the curse of dimensionality, ECOD overcomes this problem by estimating independently the density function in every dimension [2].

The *rationale* behind the ECOD detector is similar to the three-sigma rule, the detector measure how much the sample is an outlier estimating the tail of the distribution in every dimension, in this case, the assumption of independence of the dimensions is done, so in this case, the idea is to estimate d ECDF one for every dimension.

To introduce what is a ECDF the definition of CDF is needed, let $D : \mathbb{R}^n \to [0, 1]$ denote the CDF across all the n dimensions. Given s sampled points in this space such as $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_s$, naming $\mathbf{X}^{(j)}$ as the j-th entry of the vector $\mathbf{X}$ and defining $\mathbf{Y}$ as a generic random variable with the same distribution as each $\mathbf{X}_i$, then the joint

CDF for any $z \in \mathbb{R}^n$ is defined as:

$$D(z) = \mathbb{P}\underbrace{\left(Y^{(1)} \leq z^{(1)}, Y^{(2)} \leq z^{(2)}, \ldots, Y^{(n)} \leq z^{(n)}\right)}_{\mathbf{Y} \leq z} \tag{2.2}$$

This is the probability to measure how much the $\mathbf{Y}$ samples is far from the mean of the distribution in terms of the left tails. To simplifying the problem the assumption of the non dependencies of the dimensions has to be done, so a definition of $D(z)$ is:

$$D(z) = \prod_{j=1}^{d} F^{(j)}\left(x^{(j)}\right), \text{ with } z \in \mathbb{R}^n \tag{2.3}$$

Now the CDF has to be computed, giving the fact that the exact computation is not doable in this method a "data approximation" is necessary[10], the so-called Empirical Cumulative Density Function, which is defined as follows:

$$D_s(z) = \frac{\#\_of\_samples \leq z}{s} = \frac{1}{s}\sum_{i=1}^{s} I(Y_i \leq z) \tag{2.4}$$

Given the definition of ECDF in Eq 2.4 the left and right tail ECDF each dimension can be computed as follow:

$$D^{(j)}_{left}(z) = \frac{1}{s}\sum_{i=1}^{s} \mathbb{I}\left\{Y_i^{(j)} \leq z\right\}$$

$$D^{(j)}_{right}(z) = \frac{1}{s}\sum_{i=1}^{s} \mathbb{I}\left\{Y_i^{(j)} \geq z\right\}$$

Then for every Y_i the left and right probability are aggregate in the final outlier score $\mathcal{O}_i$ using the (2.6). The interesting idea of this method is that the aggregation or the use of only left or only right ECDF, does not give always the best outlier score, so the skewness coefficient for every dimension is computed and then is used to decide what ECDF has to be used for every dimensions.

The skewness coefficient for every dimension is defined in the following way:

$$\gamma_j = \frac{\frac{1}{s}\sum_{i=1}^{s}\left(Y_i^{(j)} - \overline{Y^{(j)}}\right)^3}{\left[\frac{1}{s-1}\sum_{i=1}^{s}\left(Y_i^{(j)} - \overline{Y^{(j)}}\right)^2\right]^{\frac{3}{2}}} \tag{2.5}$$

where $\overline{X^{(j)}}$ is the mean over the j-th dimension. Then the outlier score for the i-th sample $\mathcal{O}(X_i)$ can be com-

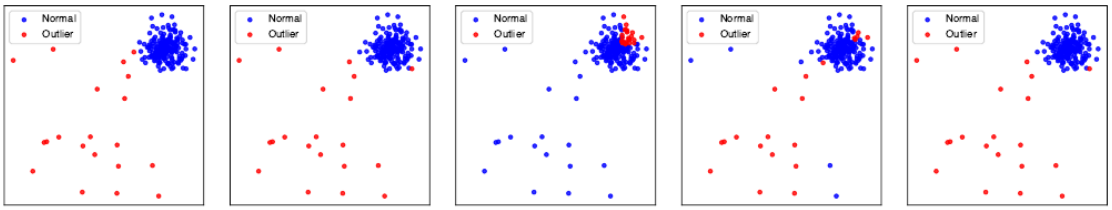
pute in the following way:

$$\mathcal{O}_{auto}(Y_i) = -\sum_{j=1}^{n} \left[\mathbb{I}\left\{\gamma_j < 0\right\} \overbrace{\log\left(D_{left}^{(j)}\left(Y_i^{(j)}\right)\right)}^{\mathcal{O}_{left}\left(Y_i^{(j)}\right)} + \mathbb{I}\left\{\gamma_j > 0\right\} \underbrace{\log\left(D_{right}^{(j)}\left(Y_i^{(j)}\right)\right)}_{\mathcal{O}_{right}\left(Y_i^{(j)}\right)} \right] \tag{2.6}$$

finally, the returned outlier score is computed as follows:

$$\mathcal{O}_i = \max\left\{ -\sum_{j=1}^{s} \mathcal{O}_{left}\left(Y_i^{(j)}\right), -\sum_{j=1}^{s} \mathcal{O}_{right}\left(Y_i^{(j)}\right), \mathcal{O}_{auto}(Y_i) \right\} \tag{2.7}$$

As we can see in 2.2 different outlier scores change the result of the methods, the problem, in this case, is that in the dataset used in this example all outliers fall on the left end of the 2−dimensional distribution, so using the left tail probabilities works surprising well[2].



(a) Ground truth (b) Left tail (c) Right tail (d) Avg of two tails (e) Skewness corrected

Figure 2.2: Those plots are taken from [2], as can be seen different tail probabilities affect in different way the results.

2.3 Autoencoder

The autoencoder-based novelty detection models are part of the so-called dimensionality reduction-based methods this type of method utilizes a reconstructive error to classify the anomalies, those methods try to find the optimal subspace where the data are encoded most compactly. Those methods assume that the features of a "normal sample" are in some way correlated and when they are projected in the optimal space the representation is compact as can be seen in Fig.1.2. The first dimensionality reduction method used is the PCA method, where the strong linearity assumption is made, meanwhile autoencoders exploit the power of non-linearity of the deep neural network to bypass this assumption.[11].

An autoencoder is a deep network architecture made up of two parts the so-called *Encoder* where an input vector $x \in \mathbb{R}^d$ is projected in a space of dimension $p < d$ which are the number of neurons in the so-called inner layer, for a deep autoencoder the number of layers to make the compression are > 1. The activation of the i-th

neuron in the hidden layer is given by:

$$h_i = f_{\vartheta}(x) = \phi\left(\sum_{j=1}^{n} W_{ij}x_j + b_i\right) \tag{2.8}$$

where ϑ are the parameters of the *Encoder* part of the network, in this case $\{W, b\}$, $b \in \mathbb{R}^p$ is the bias vector and $W \in \mathbb{R}^{p\times d}$ is the encoder weight matrix and ϕ is a non linear function called activation function, so the input vector x is mapped in a lower dimensional space.

The second part of the autoencoder is the *Decoder* in this case the network remap the hidden layer into the original space $\mathbb{R}^p \to \mathbb{R}^d$, the function for the i-th feature can be defined as follows:

$$x_i^* = g_{\vartheta^*}(h) = \phi\left(\sum_{j=1}^{n} W_{ij}^* h_j + b_j^*\right) \tag{2.9}$$

where ϑ^* are the parameters of the *Decoder* part of the network, in this case $\{W^*, b^*\}$, $b^* \in \mathbb{R}^d$ is the bias vector and $W^* \in \mathbb{R}^{d\times p}$ is the encoder weight matrix and ϕ is a non linear function called activation function, so the hidden vector h is mapped in the original space [8].

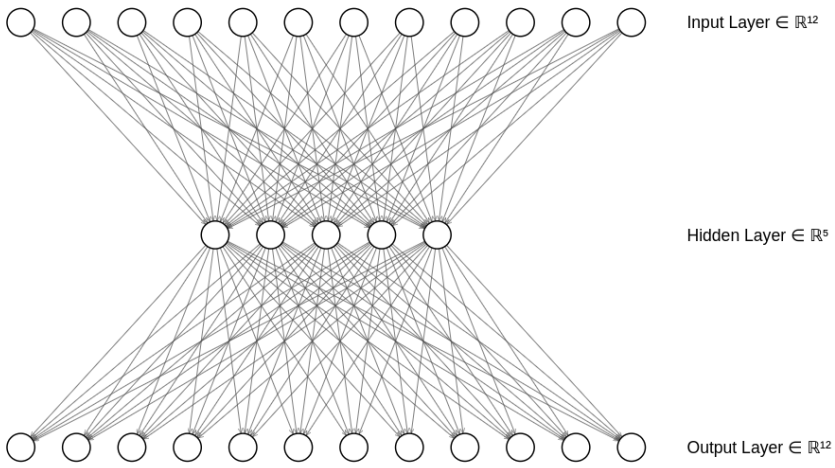


Figure 2.3: Graphical representation of an Autoencoder, created with the online tool NN SVG.

to optimize the weights and biases of the network the target (Loss or error) function has to be optimize with respect to ϑ, ϑ^*:

$$\begin{aligned}(\vartheta, \vartheta^*) &= \arg\min_{\vartheta,\vartheta^*} \frac{1}{n}\sum_{i=1}^{n} \varepsilon\left(x_i; x_i^*\right) \\ &= \arg\min_{\vartheta,\vartheta^*} \frac{1}{n}\sum_{i=1}^{n} \varepsilon\left(x_i; g_{\vartheta^*}\left(f_{\vartheta}(x_i)\right)\right)\end{aligned}$$

The target function to minimize is the so called *reconstruction error* defined as follows:

$$\varepsilon(x_i; x_i^*) = \sum_{j=1}^{d} (x_i - x_i^*)^2 \tag{2.10}$$

This type of error gives an idea on how much the reconstruction of the original sample is similar to the original one, given this property of the reconstruction error a threshold is defined so the classification of an anomalous data can be:

$$a(x_i) = \begin{cases} normal, & \varepsilon_i < \theta \\ anomalous, & \varepsilon_i > \theta \end{cases} \tag{2.11}$$

2.4 Visualization tool: *t-SNE*

For the various plots in this work the tool used for the dimensionality reduction is the *t-SNE* (t-Stochastic Neighbor Embedding) the choice of this algorithm is dictated that the fact that in high dimensional datasets preserving the local structure, on the other hand, the PCA preserve the global structures[12].

t-SNE is a non parametric method that constructs non-linear embeddings. The idea is to associate a probability distribution to the neighborhood of each data, as usual $x \in \mathbb{R}^d$ where d is the number of features, the probability is defined as follows:

$$p_{i|j} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|x_i - x_k\|^2}{2\sigma_i^2}\right)} \tag{2.12}$$

where $p_{i|j}$ can be interpreted as the likelihood that x_j is a neighbor of x_i. The free parameters of this algorithm are σ_i that are usually determined by fixing the local entropy $H(p_i)$ defined as:

$$H(p_i) = -\sum_j p_{j|i} \log_2(p_{j|i}) \tag{2.13}$$

then $H(p_i)$ is set to equal a constant across all data points $\varphi = 2^{H(p_i)}$, where φ is called perplexity[12].

The *t-SNE* construct a similar probability distribution in a low dimensional latent space, defining the projected points $x' \in \mathbb{R}^{d'}$ where $d' < d$ the definition of this new probability distribution is:

$$q_{ij} = \frac{\left(1 + \|x_i' - x_j'\|^2\right)^{-1}}{\sum_{k \neq i} \left(1 + \|x_i' - x_j'\|^2\right)^{-1}} \tag{2.14}$$

The idea is that q_{ij} is chosen to be a long tail distribution in order to preserve the short distance information and to repelling two points that are far apart in the original space. To find the parameters the loss function minimized

is the so called Kullback–Leibler divergence between q_{ij} and p_{ij}:

$$\mathcal{D}_{KL}(q||p) = \sum_{ij} p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right) \tag{2.15}$$

the minimization is done via gradient descent[12].

3

Application of Novelty detection

In the Big Data era, the detection of out-of-distribution samples *i.e.* new or anomalous pattern is becoming more and more important in everyday life and also at the Large Hadron Collider to find new physics a model-agnostic searching as to be set up[13].

3.1 Application of Novelty detection to discover new physics at the LHC

To look for new physics behind the standard model at the LHC many analyses have been carried out, but these have yet to yield statistically significant deviations from the expected background. This can indicate that no new physics is found in the data or maybe the rights signals are not inspected [13]. Another problem is the amount of data collected every second is around $1MB$ for every collision this lead to roughly $40TB$ every second, to reduce the amount of data a trigger is designed to reject the uninteresting events and keep the interesting ones, this reduces the amount of data to $200MB$ per second[14]. To search for new physics in all of these data the recent unsupervised machine learning techniques are a good choice because they are model-agnostic and they need a large amount of data. The Machine learning techniques in this field make use of low-level, high-dimensional data, meanwhile, the model used by "humans" tend to use the high-level feature with low-dimensional data[15][16][17][13].

3.2 Application of Novelty detection in economics and IoT

Data proliferation and the increasing complexity of technology have transformed the past decades over the financial market. The big tech companies now exploit the analysis of those data for better decision-making [18]. The potential value of financial big data is unlocked if it is leveraged to drive decision-making. According to a study performed by IBM in 2013 [19], 71% of banking and financial markets are leveraging big data analytics to create competitive advantages for their organizations compared to 36% in 2010, representing a huge 97% increase in just two years[18]. Therefore, traders and organizations need efficient and flawless processing of high volume, rapidly changing diverse data to generate meaningful insights and evidence-based decision-making [18][20]. This scenario given its complexity (many heterogeneous actors make, sometimes, irrational decisions) needs a complex model which tries to infer some insight into the market, this complexity paired with the rise of anomalies in the market can lead to catastrophic conclusions. The incentive of these companies to study anomalies begins with the crisis of 2007.

Anomaly detection can also be used in the fraud area, also in this case the companies are interested in detecting the anomalous pattern *i.e.* the fraud as soon as possible. Some subcategories of this area are [21]:

- **Duplication of credit card**: almost everyday business, online shopping, and electronic banking are largely dependent on them. Meanwhile, the misuse of credit or debit card is on the rise and a source of common fraud. Credit card fraud anomalies can be identified using transactional records and generally correspond to any of several indicators including unusually high payments, purchase of extremely unusual items, and high frequency of purchase. The challenge associated with detecting fraudulent transactions is detecting them in an online and unsupervised manner so that the new types of fraud are detected as soon as they happen.
- **Mobile phone**: Mobile phone fraud includes a variety of fake offers that persuade consumers to buy various products. To prevent the misuse of mobile phone accounts, it is necessary to detect any unusual usage patterns. The basic technique is to monitor the usage pattern and create a customer profile of each of the accounts.
- **Insider trading**: In recent times, investment has received renewed interest from the general public, in fields such as cryptocurrencies, NFTs and in a more conventional way the stock market. Insider trading is a criminal activity in the market, where profit is made by using inside information before it is made public. The inside information is of different forms and can affect the market's prices artificially. In this domain, early detection is important to avoid individuals or organizations from making illicit profits.

Anomaly detection can also be applied in the IoT (Internet Of Things) field, in this years more and more IoT devices have been installed in all the cities around the globe. From Amazon's vocal assistant (Alexa) to air quality monitor up to traffic light and industrial control system[22][23]. In those cases, the AD declines in different ways for example in smart cities the AD can be used in the detection of fault conditions that may lead to identifying the needs of a maintenance intervention on a service, which may lead to saving money or even, in some cases, life[24]. Meanwhile in the industries AD can be used to make that the produced samples do not have faults or malfunction of industrial assembly line[25].

4 Dataset

The idea behind this work is to design a novelty detection pipeline a pictorical representation can be seen in Fig. 1.1 is divided into three experiments with different characteristics. In this chapter, the definition of these three datasets and a small introduction to the experiments is given.

4.1 Contamination rate

In order to understand the creation of the datasets the meaning of *contamination rate* has to be defined .
Given a set of samples $\mathcal{D}$ of cardinality d and considering another set $\mathcal{A}$ of cardinality a, defining the set $\mathcal{U}$ as the union of $\mathcal{D}$ and $\mathcal{A}$, using mathematics $\mathcal{U} = \mathcal{A} \cup \mathcal{D}$, with cardinality $u = a + d$.
The *contamination rate* of the new set can be defined as:

$$p = \frac{a}{u} \tag{4.1}$$

so the contamination rate is the percentage of anomalies in the set $\mathcal{U}$.

4.2 MNIST

All the created datasets originated from the MNIST handwritten digits database, a large dataset used in most machine learning courses as a so-called toy dataset, *i.e.* a pretty simple dataset used to validate the models in the first phases of the ML workflow. It is composed of grayscale images 28×28 pixels, the cardinality of the training set is $60'000$, meanwhile, the cardinality of the test set is $10'000$, and of course, the number of classes is 10.

4.2.1 First dataset: Anomalized MNIST

The purpose of the first experiment was to give an insight into the internal representation of a NN of the anomalous samples, in other words, to inspect the *latent space* to see if a space like the one in Fig.1.2 can be reconstructed by an NN trained with anomalous data. In this case, the anomalies are injected both in the training set and in the test set, so in this case, the correct name is not novelty detection, but anomaly detection. The anomalous data, in this case, are the digits 0 and 1, the other digits are the normal samples. The structure of the data in this experiment is preserved, *i.e.* the images are not flattened. To compute the number of anomalies to inject in the dataset the *rationale* is always the same, defining u the cardinality of the set $\mathcal{U} = \mathcal{X} \cup \mathcal{X}_{anomalous}$, given the cardinality of the two subsets $|\mathcal{X}| = x$ and $|\mathcal{X}_{anomalous}| = x_A$, the following set of equation holds:

$$\begin{cases} u = x + x_A \\ x_A = up \end{cases} \tag{4.2}$$

solving for x_A the following expression is obtained:

$$x_A = \frac{p}{1-p}x \tag{4.3}$$

Logically the *contamination rate* has a theoretical upper limit which is 50% in this case the anomalies are more than half of the overall dataset, on the other hand, there is also another upper bound, this is given by the cardinality of the classes in the MNIST dataset which are listed in Appendix A.1. The total number, without the 0 and 1, in the training set is 41377 and in the test set is 6853, the cardinality of the classes 0 and 1 summed up is 14780, so the *real contamination rate* upper limit is:

$$p_{real} = \frac{14780}{41377 + 6853} = \frac{14780}{48230} \sim 0.31 \tag{4.4}$$

In this experiment the *contamination rate* used are: $[0.01, 0.02, 0.05, 0.1, 0.2, 0.25]$. To study different "types of novelty" different "misclassification" were made, in the first case named *OAL* (**O**ne **A**nomalies **L**abel) all the anomalies are classified as a 9, in the second case named *RAL* (**R**andom **A**nomalies **L**abels) at every anomaly was assigned a random label from 2 to 9, the former mimics a pattern pretty similar to an already known one, on the other hand, the latter mimics a completely unknown pattern, in this case, the label is assigned randomly.

4.2.2 Second dataset: Tabularized MNIST

The purpose of the second experiment was to apply the same idea of the first experiment with two differences:

- The anomalous data are injected only in the test set.
- The images are flattened to mimic the data used in the economic field, *i.e.* tabular data.

Tabular data

Tabular data refers to data that is organized in a table with rows and columns, this type of data can give a more informative space in the economic field instead of time series, in some way tabular data are multivariate time series. On the other hand given the informative power and complexity of this multidimensional time series to unlock their full potential deep-learning techniques are the best type of model to be used [26].

This transformation of the images in tabular data brings up an interesting problem in Sec. 4.2.1 the dataset has a so-called "internal structure", *sc.* the position of the pixels, in the case of "tabularize MNIST" the structure is no more definite and in theory, the space is noisier and it is more difficult to make classification on it. In this case the division of the dataset in *OAL* and *RAL* is not done, because the only difference between the two are the labels in the test set, so the classifier does not take in account the different labelling in the training phase.

Given the fact that the anomalies are injected only in the test set in this case the *real contamination rate* is:

$$p_{real} = \frac{14780}{6853} \sim 2.2 \tag{4.5}$$

so the anomalies are more than the non-anomalous samples and the *theoretical upper limit* of 0.5 *contamination rate* can be applied, in the datasets created with the tabular data the *contamination rates* used are :$[0.01, 0.05, 0.1, 0.2, 0.3, 0.4]$.

4.2.3 Third dataset: Pairwise dataset

To mimic a pairwise asset ranking the "tabularized MNIST" was divided in pair by the digits, *i.e.* $[0, 1], [0, 2], \ldots, [8, 9]$. The number of couple can be easily computed using binomial factor:

$$C_{n,k} = \binom{n}{k} = \frac{n!}{k!(n-k)!} \tag{4.6}$$

in this case $k = 2$ and $n = 10$:

$$C_{10,2} = \binom{10}{2} = \frac{10!}{2!8!} = 45 \tag{4.7}$$

given the large number of couples, the plots presented in this thesis are the ones of the couple $[0, 1]$, but the results are general. As for the dataset presented in Sec 4.2.2 the anomalies are injected only in the test set. Given the nature of the problem of asset ranking the "type" of labelling chosen in this dataset was the RAL one.

5 Experiments

5.1 Experiment 1: How anomaly affect the inference

In this preliminary experiment, the research question is: *How do the anomalies affect the training and the inference of an artificial neural network using image data?* The steps of this experiment can be summarised as follows:

1. Creation of the dataset.
2. Modelling a Convolutional Neural Network that makes inference on the 8 classes.
3. Project the various layers of the CNN using *t-SNE* algorithm (Sec. 2.4).
4. Repeat from 1 to 3 varying the *contamination rate*.

For this type of data, the more suitable architecture is the CNN which can easily handle the images and take advantage of the internal features of this data such as locality and translation invariance.

5.1.1 The CNN Architecture

Two kinds of basic layers make up a CNN: a convolution layer that computes the convolution of the input with a bank of filters, and pooling layers that coarse-grain the input while maintaining locality and spatial structure. For two-dimensional data, a layer l is characterized by three numbers: height H_l, width W_l , and depth D_l. The height and width correspond to the sizes of the two-dimensional spatial (W_l, H_l)-plane, and the depth D_l to the number of filters in that layer. All neurons corresponding to a particular filter have the same parameters[12]. The architecture created for this experiment is summarized in Fig. 5.1.

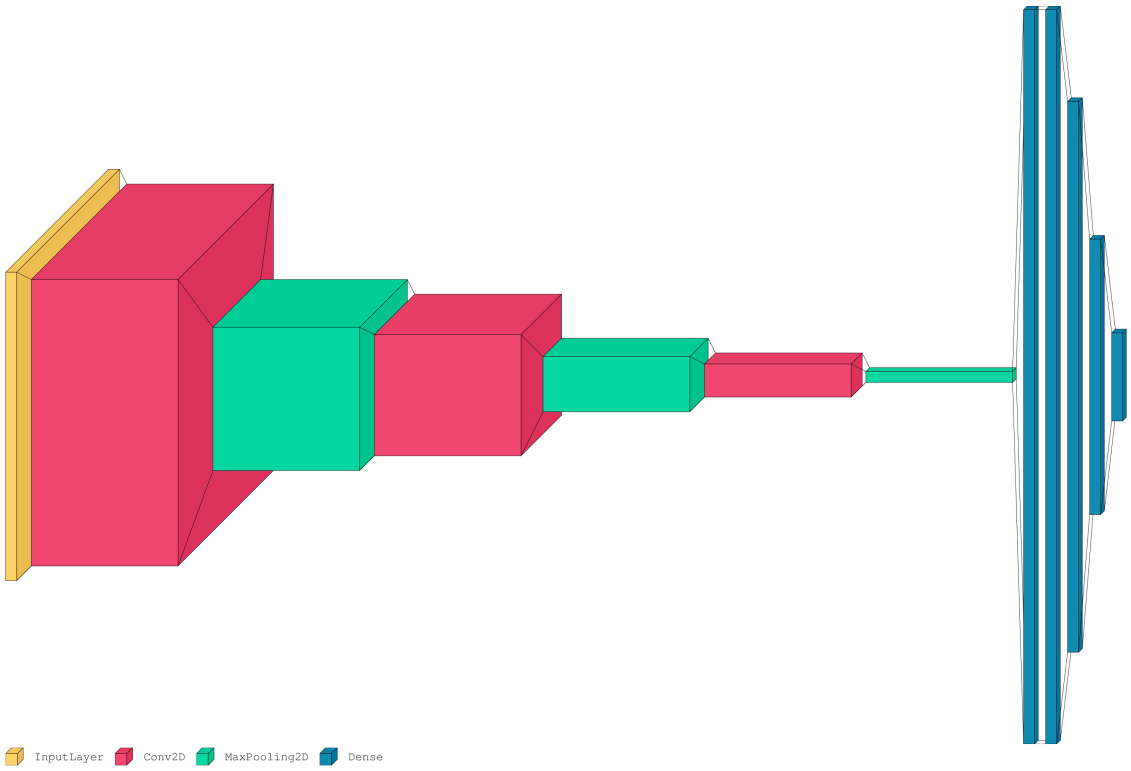


Figure 5.1: Graphical representation of the architecture used in the Experiment 1 to make the inference on the dataset with anomalies, the plot is created using [3].

5.1.2 Projection using *t-SNE* of *OAL* dataset

At the end of the training for every *contamination rate* the last two `Dense` layers are projected using different perplexity. The projections are almost identical so only the second last layer plots are shown. In the following plots the bigger dots are the anomalies meanwhile the little are the anomalous ones, the hues are given by the classes label.

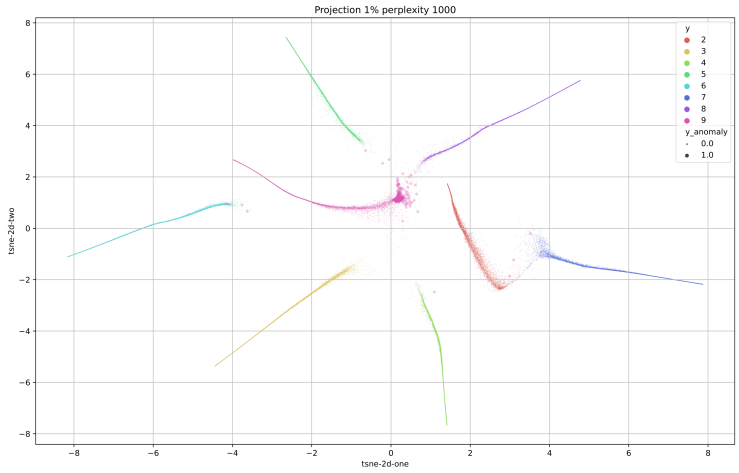


Figure 5.2: OAL dataset with *contamination rate* $= 0.01$ processed by the CNN and projected in the second-last layer using *t-SNE*, in this configuration the density of the anomalies is too low to form a distinct cluster from the 9 class.

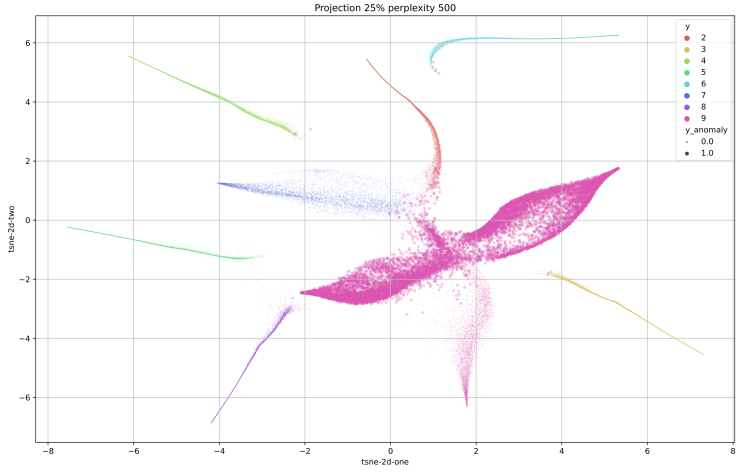


Figure 5.3: OAL dataset with *contamination rate* $= 0.25$ processed by the CNN and projected in the second-last layer using *t-SNE*. In this configuration can be clearly see that the class of number 9 has became a so-called superclass (union of more than one class), where three different ”wings” can be seen.

5.1.3 Projection using *t-SNE* of *RAL* dataset

The dataset, in this case, is more complex, because the labels of the anomalies are not assigned to a fixed label. This type of labeling make the inference and the training of the neural network more difficult because the labels are more noisy.

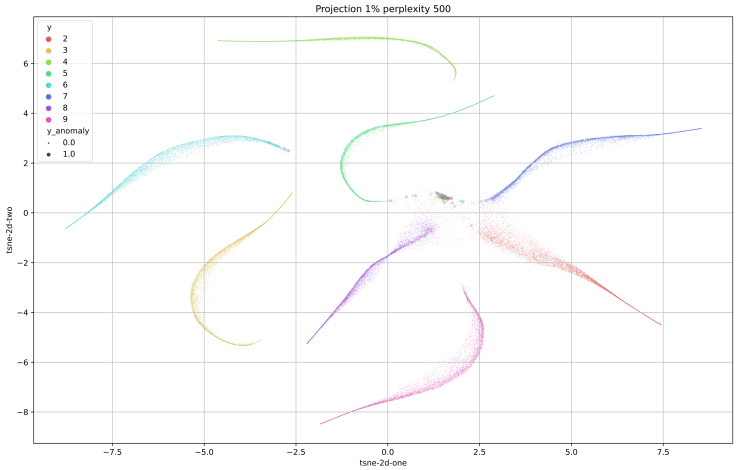


Figure 5.4: RAL dataset with *contamination rate* $= 0.01$ processed by the CNN and projected in the second-last layer using *t-SNE*. In this configuration the cluster of anomalies is pretty separated from the other classes.

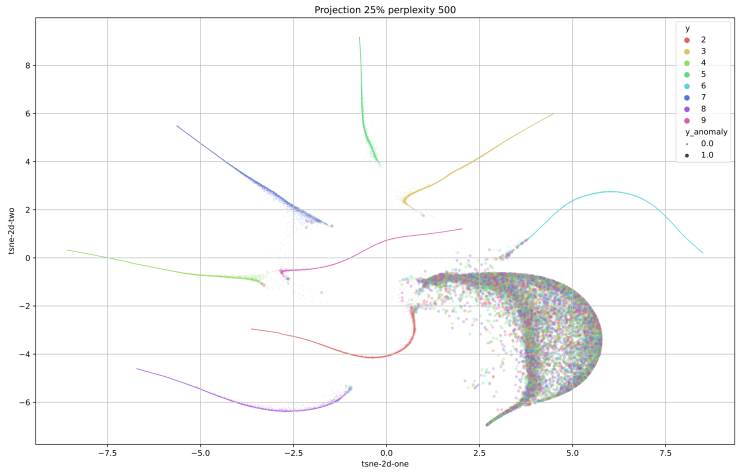


Figure 5.5: RAL dataset with *contamination rate* $= 0.25$ processed by the CNN and projected in the second-last layer using *t-SNE*. As in Fig.5.3 also in this configuration an anomalous cluster it is distinct from the other classes.

5.2 Experiment 2: How anomalies affect the inference in tabular data

In this second experiment, the research question is: *How do the anomalies affect the training and the inference of an artificial neural network using tabular data?* The steps of this experiment are pretty similar to the previous one (Sec.5.1) and can be summarised as follows:

1. Creation of the dataset.
2. Modelling a FNN that makes inference on the 8 classes.
3. Project the various layers of the FNN using *t-SNE* algorithm (Sec. 2.4).
4. Compute the accuracy over the non-anomalous, anomalous and all the samples in the test set.
5. Repeat from 1 to 4 varying the *contamination rate*.

With this dataset, the structure of the data, the locality of the pixels and the translation invariance are lost given the flattenization of the images, so in this case, the only suitable type of ANN is the MLP.

5.2.1 Multi Layer Perceptron

The MLP used in this experiment is pretty simple and a graphical representation is shown in Fig. 5.6,

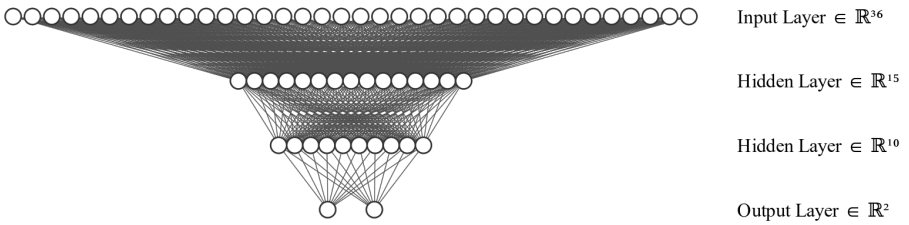


Figure 5.6: Graphical representation of the MLP architecture used in the Experiment 2 to make the inference on the tabular dataset with anomalies.

5.2.2 Projection and accuracy study on the tabular *OAL* dataset

The projection, as in Sec. 5.1.2 and 5.1.3, is done with the *t-SNE* algorithm, in this case the projections are made only the test set, because the anomalous data are injected only in it. In this type of datasets the spatial correlation of the pixels is lost so the result of the projection is less clear.

Figure 5.7: Test set of the OAL tabular dataset with *contamination rate* $= 0.05$ processed by the ANN and projected in the last layer using *t-SNE*. In this configuration the anomalies are not clustered near the class 9, this is due to the fact that the ANN tends to classify those samples as 8. This can also be seen in the analysis of the analysis of the accuracies of the ANN in Fig.5.9.

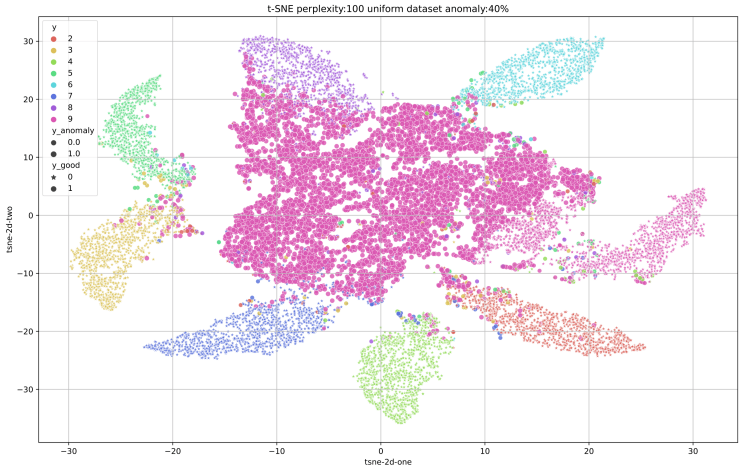


Figure 5.8: Test set of the OAL tabular dataset with *contamination rate* $= 0.40$ processed by the ANN and projected in the last layer using *t-SNE*. In this configuration the anomalous cluster can be seen in the center of the projected latent space.

To study how the ANN classify the anomalous sample the ”accuracies” of the anomalies in the test set are calculated. As can be seen in Fig. 5.9 the anomalies tend to be classified as 8. In the second configuration, the one

shown in Fig. 5.10, given the large number of anomalous samples, the classifier tends to classify the samples the majority of the times as 8 but also as 7 and as 1.

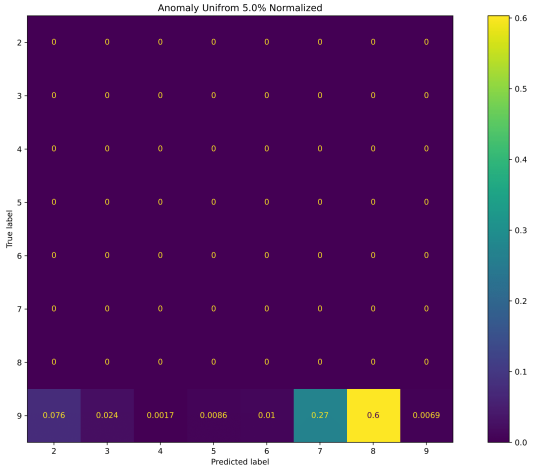


Figure 5.9: Accuracies computed on the anomalies of test set with *contamination rate* of 0.05.

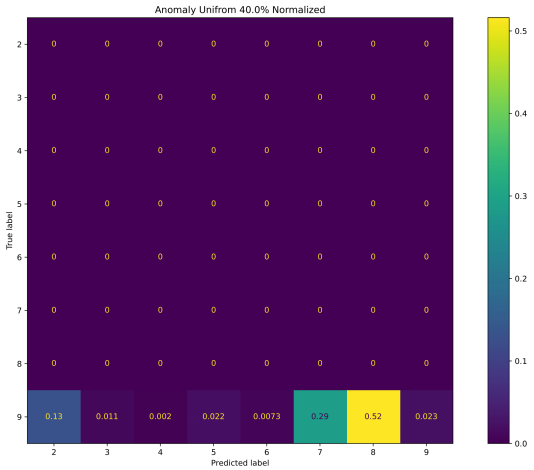


Figure 5.10: Accuracies computed on the anomalies of test set with *contamination rate* of 0.40.

Then a sanity check on the classifier is done to confirm that the ANN is trained correctly as can be seen in Fig.5.11 and Fig.5.12 the accuracies are between 94% and 99%.

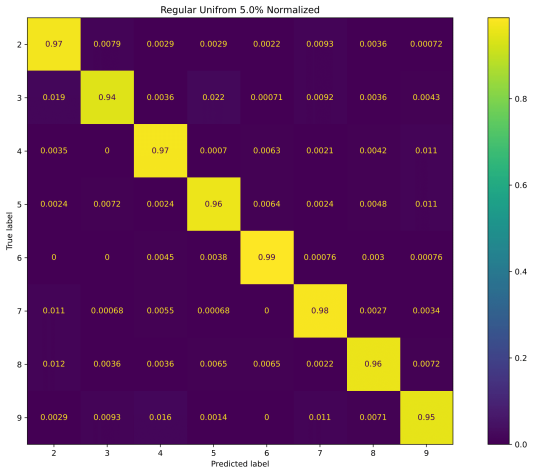


Figure 5.11: Accuracies computed on the test set without the anomalous samples with *contamination rate* of 0.05.

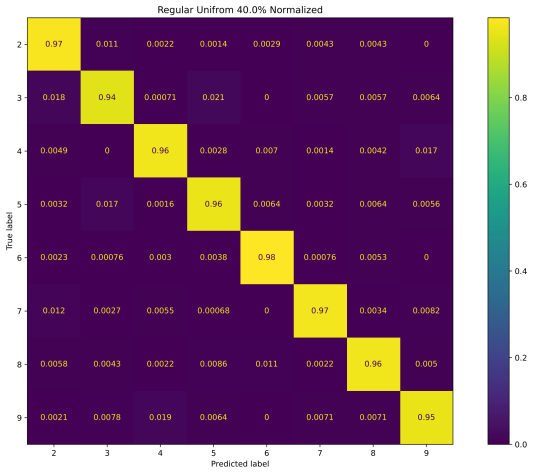


Figure 5.12: Accuracies computed on the test set without the anomalous samples with *contamination rate* of 0.40.

5.3 Experiment 3: Novelty detection pipeline

Given the interesting results of the former experiments (Sec.5.1 and Sec.5.2) in this one the research question is: *The embeddng of an ANN can be used to improve the performances of an anomaly detections algorithm?*. The high-level idea is to pair a classifier already trained, which has to be left untouched, with an anomaly detector and

to reduce the complexity and computational time of the novelty detector, *i.e.* using a small space than the input space.

5.3.1 Classifiers used in experiment 3

Two different ANNs were considered in the third experiment. The one already presented in Sec.5.2.1, from now on `Dense15`, and another ANN presented in Fig.5.13, where the AD was applied to the input and the inner layers of the two NN.

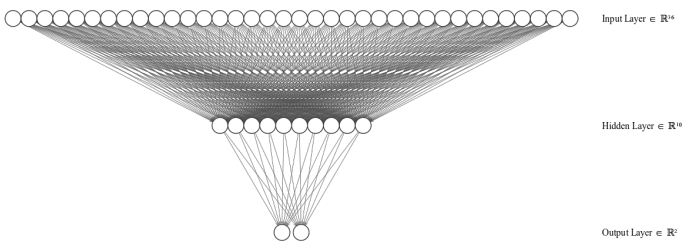


Figure 5.13: Graphical representation of the `Dense10` architecture.

5.3.2 Projection of pair 01 using *t-SNE*

Given the small number of anomalous and normal samples in this experiment the anomalous clusters are less clear, in Fig.5.14 only the clusters of the normal samples are distinguishable.

In the following plot the anomalous samples form a well defined cluster, separated from the two "normal clusters":

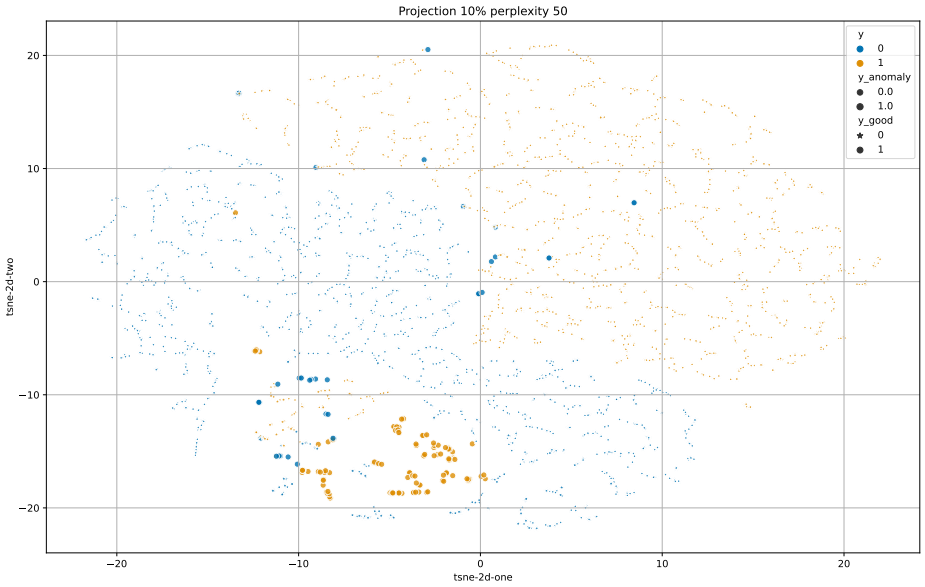


Figure 5.14: Pair 01 in a dataset with *contamination rate*$=0.01$ processed by `Dense15` and projected in the last layer using *t-SNE*.

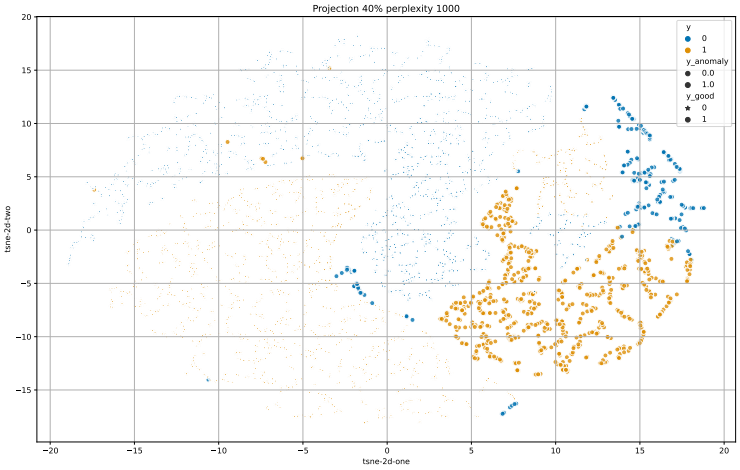


Figure 5.15: Pair 01 in a dataset with *contamination rate*$=0.4$ processed by `Dense15` and projected in the last layer of the architecture `Dense15` using *t-SNE*.

To make the anomalous cluster more definite the *contamination rate* can be increased, in the following plot

(with *contamination rate* $= 0.6$) the three clusters are well separated, so a topological difference in the latent space can be impute in this case.

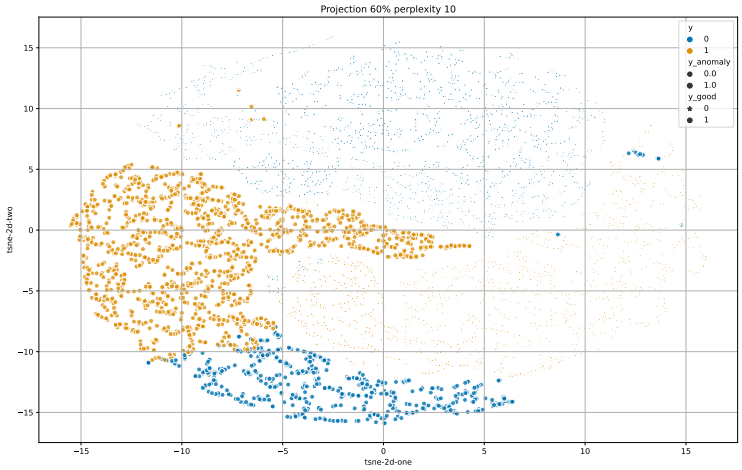


Figure 5.16: Pair 01 in a dataset with *contamination rate*$= 0.6$ processed by `Dense15` and projected in the last layer of the architecture `Dense15` using *t-SNE*.

5.3.3 Evaluation metrics for AD

To study the effectiveness of the AD algorithm used in this experiment a digression on the metrics has to be done. The most popular metrics used to compare performances are F1-score, Area Under the Receiver Operating Characteristic (ROC) Curve (AUC) and Average Precision score (AVPR). The F1-score and AVPR are highly sensitive to the *Contamination rate*[27], this is a problem because this scores can be artificially increased by tuning the *contamination rate*, so for this work, the metric chosen is AUC.

A common evaluation metric in pairwise is the *Confusion Matrix* a special type of 2 dimensional contingency matrix, where each row represents an instance of a class, in this case, the matrix can be visualized in the following way:

True Positive *tp*	False Negative *fn*
False Positive *fp*	True Negative *tn*

The ROC is a graph showing the performance of a classification model, it show two parameters the True Positive Rate (TPR) and the False Positive Rate (FPR), their mathematical definitions are:

$$\begin{cases} TPR = \frac{TP}{TP+FN} \\ FPR = \frac{FP}{FP+TN} \end{cases} \tag{5.1}$$

where TP are the true positive, FN the false negative and TN are the true negative. A graphical representation of the ROC curve can be:

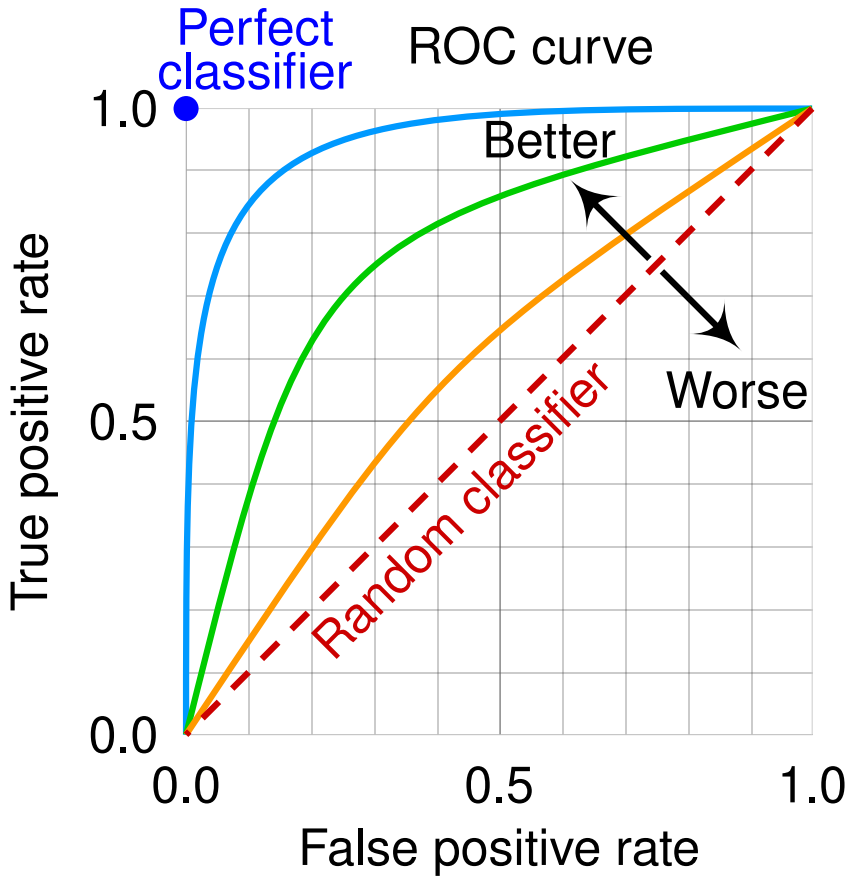


Figure 5.17: A graphical representation of the ROC curve

The AUC is simply the area under the ROC curve.

5.3.4 Anomaly Detection on the embeddings

In this part of the third experiment, the aim is to apply the anomaly detector to the input set and the various outputs of the ANN, so the step can be summarized in the following steps:

1. Train the ANNs.
2. Freeze the weights.
3. Collect the samples, in all the layers.
4. Train the anomaly detectors and apply them.
5. Compute the AUC.
6. Repeat from 1 to 5 for all the pairs.

The previous steps apply both to the `Dense 10` and `Dense15` architecture.

Experiment 3 on the `Dense10` architecture

The architecture used in this sub-experiment is the one shown in Fig.5.13, so the NN is composed of three layers, for every pair and every layer every anomaly detector algorithm is optimize and then trained. Given the fact that the AUC is resilient to the contamination rate only the results aggregated per *contamination rate* $= 0.1$ are shown in the following bar plot.

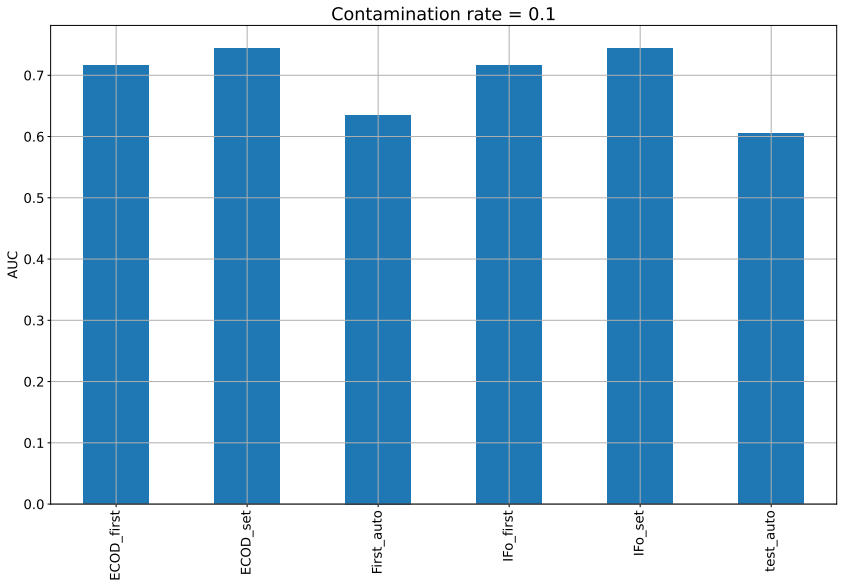


Figure 5.18: Mean of AUC for the pairwise dataset with *contamination rate* 0.1, on the x axis the name of the anomaly detector are shown and on the y axis the AUC.

The mean of the AUC is greater in the case of Isolation forest applied on the input set so in this particular case, the embedding seems not very informative. To prove that the AUC is agnostic in respect of the *contamination rate* the mean and standard deviation of them on all the *contamination rate* list used in this work is done, the results are summarized in Tab.5.1, as can be seen in this case the standard deviation is of order 10^{-3}, two order of magnitude below the mean.

Layer and AD algorithm	mean of AUC	standard deviation of the AUC
ECOD input layer	0.743	0.004
ECOD first layer	0.720	0.004
Autoencoder input layer	0.6043	0.004
Autoencoder first layer	0.637	0.003
IF input layer	0.744	0.004
IF first layer	0.730	0.004

Table 5.1: AUC scores of the various anomaly detectors on the layers of `Dense10` neural network.

Experiment 3 on the `Dense15` architecture

The steps of this sub-experiment are the same as the previous one, the `Dense15` architecture is composed of three layers (input, first layer and second layer), meanwhile the `Dense10` is composed of two layer.

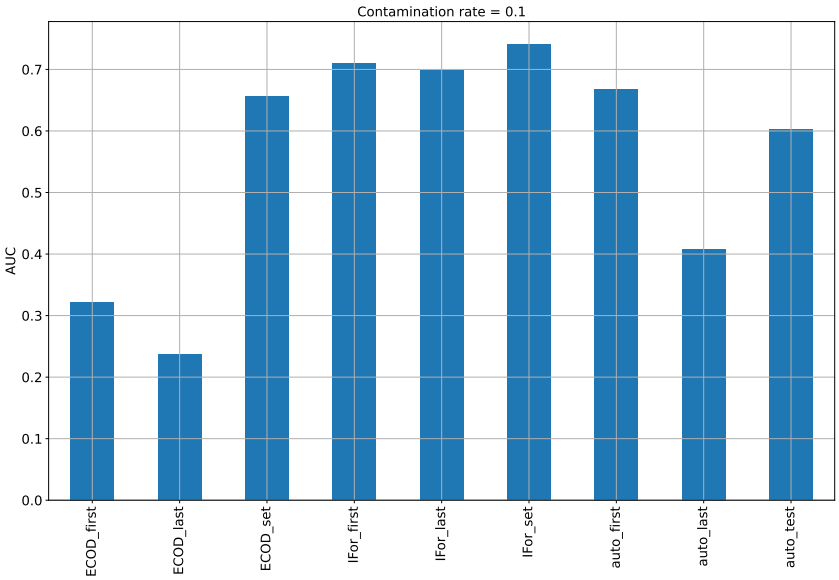


Figure 5.19: Mean of AUC for the pairwise dataset with *contamination rate* 0.1, on the x axis the name of the anomaly detector are shown and on the y axis the AUC.

In this experiment the isolation forest applied on the input layer outperform the others AD.

Layer and AD algorithm	mean of AUC	standard deviation of the AUC
ECOD input layer	0.62	0.03
ECOD first layer	0.29	0.03
ECOD second layer	0.22	0.01
Autoencoder input layer	0.605	0.001
Autoencoder first layer	0.6740	0.003
Autoencoder second layer	0.406	0.004
IF input layer	0.744	0.002
IF first layer	0.718	0.005
IF second layer	0.686	0.01

Table 5.2: AUC scores of the various anomaly detectors on the layers of the `Dense15` neural network.

6

Conclusion and future perspectives

This thesis is an overview of anomaly detection algorithms on a multi (high) dimensional space. Given the generality of the data (no assumptions are made) this type of algorithm can be applied in all the fields cited in Cap. 3. The scientific review part was a fundamental step, the study of many papers helped to understand and to select the three algorithms applied from all the ones which can be found in the literature. The three AD algorithms proposed in this work are completely unsupervised. This aspect is critical since in the field where those have to be applied (economy) the labels of the anomalous samples do not exist.

The work concentrates on the application of anomaly detection algorithms on tabular data, this type of data is widely used in all the fields of science from biology to physics up to economy. The principal problem of this study is the size of the datasets created. In experiment 3 the number of datasets used is 270 (45 pairs and 6 contamination rates). The training of the various algorithm on this number of datasets is computationally demanding and a non trivial optimization challenge. Some final thoughts on the best algorithm found in this work have to be done. Why does Isolation Forest outperform the other algorithms? This is due to some factors which are intrinsic in its definition. In comparison to the most used algorithms in the literature, the ANN, the Random Forest does not struggle with uninformative features, which are intrinsically encoded in tabular data [28]. Also, the time needed to train and optimize this type of algorithm is shorter with respect to the Neural Networks. A GPU is not needed for the tree-based models, also tree-based algorithms are interpretable and not a *black box* such as the neural networks. The next steps of this work are to increase the complexity of the dataset using for example *CIFAR100*, a subset of *ImageNet* and the study of the interpretable parameters in the Isolation Forest.

References

[1] F. T. Liu, K. Ting, and Z.-H. Zhou, "Isolation forest," 01 2009, pp. 413 – 422.

[2] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, and G. Chen, "Ecod: Unsupervised outlier detection using empirical cumulative distribution functions," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2022.

[3] P. Gavrikov, "visualkeras," https://github.com/paulgavrikov/visualkeras, 2020.

[4] M. A. F. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "Review: A review of novelty detection," *Signal Process.*, vol. 99, p. 215–249, jun 2014. [Online]. Available: https://doi.org/10.1016/j.sigpro.2013.12.026

[5] S. Thudumu, P. Branch, J. Jin, and J. J. Singh, "A comprehensive survey of anomaly detection techniques for high dimensional big data," *Journal of Big Data*, vol. 7, 12 2020.

[6] R. Bellman, *Dynamic programming.*, reprint of the sixth (1972) edition ed. Mineola, NY: Dover Publications, 2003.

[7] R. J. Schalkoff, *Pattern Recognition: Statistical, Structural and Neural Approaches*. USA: John Wiley & Sons, Inc., 1991.

[8] A. L. Alfeo, M. G. Cimino, G. Manco, E. Ritacco, and G. Vaglini, "Using an autoencoder in the design of an anomaly detector for smart manufacturing," *Pattern Recognition Letters*, vol. 136, pp. 272–278, Aug. 2020. [Online]. Available: https://doi.org/10.1016/j.patrec.2020.06.008

[9] Z. Ding and M. Fei, "An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window," *IFAC Proceedings Volumes*, vol. 46, no. 20, pp. 12–17, 2013, 3rd IFAC Conference on Intelligent Control and Automation Science ICONS 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667016314999

[10] C. P. T. Kandethody M. Ramachandran, *Mathematical Statistics With Applications in R*, 3rd ed. Academic Press, 2021;2020.

[11] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, "Autoencoder-based network anomaly detection," in *2018 Wireless Telecommunications Symposium (WTS)*, 2018, pp. 1–5.

[12] P. Mehta, M. Bukov, C.-H. Wang, A. G. Day, C. Richardson, C. K. Fisher, and D. J. Schwab, "A high-bias, low-variance introduction to machine learning for physicists," *Physics Reports*, vol. 810, pp. 1–124, may 2019. [Online]. Available: https://doi.org/10.1016%2Fj.physrep.2019.03.001

[13] L. Bradshaw, S. Chang, and B. Ostdiek, "Creating simple, interpretable anomaly detectors for new physics in jet substructure," 2022. [Online]. Available: https://arxiv.org/abs/2203.01343

[14] L. Evans, Ed., *The Large Hadron Collider: A marvel technology*, 2009.

[15] M. C. Romão, N. F. Castro, and R. Pedro, "Finding new physics without learning about it: anomaly detection as a tool for searches at colliders," *The European Physical Journal C*, vol. 81, no. 1, jan 2021. [Online]. Available: https://doi.org/10.1140%2Fepjc%2Fs10052-020-08807-w

[16] T. Finke, M. Krämer, A. Morandini, A. Mück, and I. Oleksiyuk, "Autoencoders for unsupervised anomaly detection in high energy physics," *Journal of High Energy Physics*, vol. 2021, no. 6, jun 2021. [Online]. Available: https://doi.org/10.1007%2Fjhep06%282021%29161

[17] J. Collins, K. Howe, and B. Nachman, "Anomaly detection for resonant new physics with machine learning," *Physical Review Letters*, vol. 121, no. 24, dec 2018. [Online]. Available: https://doi.org/10.1103%2Fphysrevlett.121.241803

[18] M. Ahmed, N. Choudhury, and S. Uddin, "Anomaly detection on big data in financial markets," in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, ser. ASONAM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 998–1001. [Online]. Available: https://doi.org/10.1145/3110025.3119402

[19] M. Schroeck, R. Shockley, J. Smart, D. Romero-Morales, and P. Tufano, "Analytics: The real-world use of big data," IBM Institute for Business Value, IBM Institute for Business Value - Executive Report, 2012.

[20] A. Gandomi and M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management*, vol. 35, pp. 137–144, 04 2015.

[21] M. Ahmed, A. N. Mahmood, and M. R. Islam, "A survey of anomaly detection techniques in financial domain," *Future Generation Computer Systems*, vol. 55, pp. 278–288, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X15000023

[22] N. Díaz, J. Guerra, and J. Nicola, "Smart traffic light control system," in *2018 IEEE Third Ecuador Technical Chapters Meeting (ETCM)*, 2018, pp. 1–4.

[23] A. L. Alfeo, M. G. Cimino, G. Manco, E. Ritacco, and G. Vaglini, "Using an autoencoder in the design of an anomaly detector for smart manufacturing," *Pattern Recognition Letters*, vol. 136, pp. 272–278, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865520302269

[24] P. Bellini, D. Cenni, P. Nesi, and M. Soderi, "Anomaly detection on iot data for smart city," in *2020 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2020, pp. 416–421.

[25] S. Zhang, F. Ye, B. Wang, and T. G. Habetler, "Semi-supervised learning of bearing anomaly detection via deep variational autoencoders," 2019. [Online]. Available: https://arxiv.org/abs/1912.01096

[26] I. Padhi, Y. Schiff, I. Melnyk, M. Rigotti, Y. Mroueh, P. L. Dognin, J. Ross, R. Nair, and E. Altman, "Tabular transformers for modeling multivariate time series," *CoRR*, vol. abs/2011.01843, 2020. [Online]. Available: https://arxiv.org/abs/2011.01843

[27] D. Fourure, M. U. Javaid, N. Posocco, and S. Tihon, "Anomaly detection: How to artificially increase your f1-score with a biased evaluation protocol," 2021.

[28] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on tabular data?" 2022. [Online]. Available: https://arxiv.org/abs/2207.08815

A Appendix

A.1 MNIST dataset classes cardinality

Classes	Cardinality_train	Cardinality_test
0	5928	980
1	6742	1135
2	5958	1032
3	6131	1010
4	5842	982
5	5421	892
6	5918	958
7	6265	1028
8	5851	974
9	5946	1009