Anomaly detection using Contrastive Predictive Coding
Advanced Deep Learning Models and Methods 2022 Project

Amedeo Carrioli
Politecnico di Milano
amedeo.carrioli@mail.polimi.it

Greta Corti
Politecnico di Milano
greta.corti@mail.polimi.it

Giacomo Boracchi
Politecnico di Milano
giacomo.boracchi@polimi.it

Matteo Matteucci
Politecnico di Milano
matteo.matteucci@polimi.it

Abstract

Due to the lack of labeled data, building reliable models that make detection of anomalies is challenging. To solve this problem, contrastive learning approaches are becoming increasingly popular, given the impressive results they have achieved in self-supervised representation learning environments. In this paper, we exploit the Contrastive Predictive Coding model [11], an unsupervised learning method which extracts useful representations from high-dimensional data, and its patch-wise contrastive loss which can directly be interpreted as an anomaly score [7]. The resulting model has been implemented using three different well-known and promising models such as ResNet18, ResNet50, and DenseNet [10]. We test and compare these models on some classes of the challenging MVTec-AD dataset.

1. Introduction

An anomaly is an observation that differs significantly from the vast majority of the data. Anomaly detection tries to identify unexpected samples which do not conform to a well defined notion of normal behaviour. Anomaly detection is a crucial point in many applications, mostly in real-time scenarios, where spotting anomalies is vital such as in healthcare, security applications and automated industrial inspections.

Our project consists in replicating the anomaly detection model presented in the paper [7] which extends paper [11] where is described the structure and the functioning of the Contrastive Predictive Coding (CPC). We, then, extend this anomaly detection model adding some baseline encoders, such as ResNet50 and DenseNet [10], to study if there is any improvement on detecting anomalies.

Firstly, we started to implement the paper about CPC to better understand the mechanism behind this method: it is composed by an encoder which compresses high-dimensional data into a much more compact latent embedding space and an auto-regressive model that makes predictions many steps in the future, and we rely on Noise-Contrastive Estimation for the loss function. Next, we added the part about anomaly detection omitting the auto-regressive model and implementing the CPC contrastive loss as anomaly score, and, at last, we added the new two encoder models: ResNet50 and DenseNet, besides the one described in the paper.

2. Related work

In this section, we present an overview of contrastive learning methods, anomaly detection models and residual neural networks.

2.1. Contrastive Learning Methods

Contrastive learning is a machine learning technique who's goal is to learn the general features of a dataset without labels by teaching the model which data points are similar or different, hence the term, self-supervised learning. This technique has achieved impressive results [11] [4]. Depending on the domain and downstream task, different transformations can be chosen. For example, on image data, random data augmentation such as random cropping has proven useful [4]. In this project, we use the Contrastive Predictive Coding model [11] [9], which makes use of temporal transformations. Generally, these kind of approaches are evaluated by training a linear classifier on top of the created representations and by measuring the performance that this linear classifier can achieve on downstream tasks.

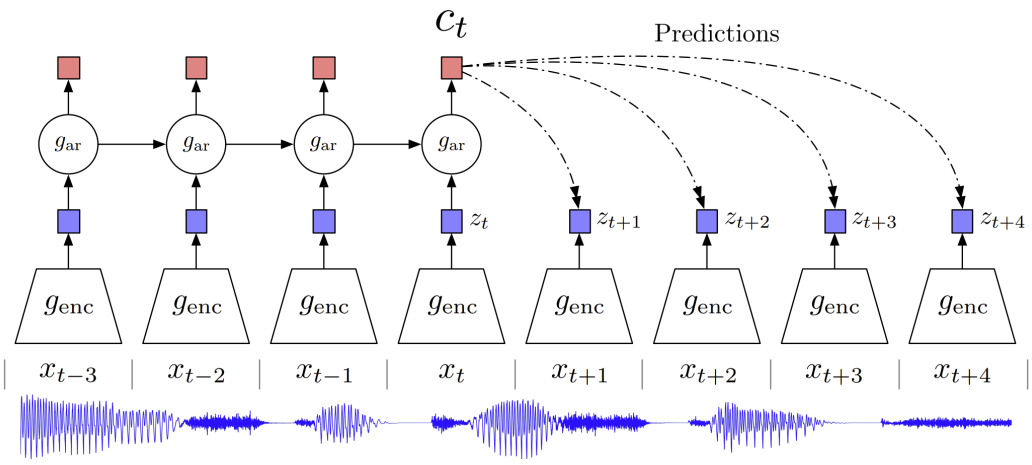

Figure 1. Overview of Contrastive Predictive Coding

2.2. Anomaly Detection Models

As we have already said, anomaly detection (AD) is the process of identifying unexpected items or events in data sets, which differ from the norm. Three broad categories of anomaly detection techniques exist: supervised, semi-supervised, and unsupervised [3]. Supervised anomaly detection techniques demand a data set with a complete set of normal and anomalous labels for a classification algorithm to work with. Semi-supervised methods assume that the training data has labeled samples only for the normal class. Unsupervised methods are the most widely applicable because they do not require training data. Models in this category make the implicit assumption that normal data are more frequent than anomalies in test data.
Several previous works investigate the use of contrastive learning for AD. The SimCLR framework [4] is applied in many works [14] to learn representations of the data which are used to calculate a density estimation that is used as an anomaly score. Another comparable contrastive learning anomaly detection method is NeuTraL AD [12]. In contrast to our method, this approach has been evaluated on time-series and tabular data, but this is the only difference since it makes use of a contrastive loss with learnable transformations and reuses this loss as anomaly score.

2.3. Residual Neural Networks

Residual Network (ResNet) is a deep learning model used for computer vision applications. It is a Convolutional Neural Network (CNN) architecture designed to support hundreds or thousands of convolutional layers. ResNet reduces significantly the problem of the vanishing gradient caused by adding many layers to the network using the skip connections [8]. As a matter of fact, ResNet was not the first network to make use of shortcut connections. The authors of a study on Highway Network [13] also introduced gated shortcut connections; however, experiments show that the highway network does not perform better than ResNet. Therefore, ResNet quickly became one of the most popular architectures used because of its compelling results. In [8] networks with different number of deep layers have been tested to show the performance: ResNet with 18, 34, 50, 101, and more layers exist in the literature. This network has gained so much popularity that its architecture has been studied heavily giving birth to new architectures and variants: ResNeXt [15], DenseNet [10], Wide ResNet [16]
DenseNet has improved the state of the art in many datasets [10] and it's another promising architecture that still uses connections between layers like Resnets (reducing the vanishing gradient problem) but also uses less parameters and has the peculiarity that the output of each convolutional layer is forwarded and used as input to every convolutional layers that comes after.

3. Proposed approach

We will now describe the proposed model in details: in the first subsection, we will present Contrastive Predictive Coding in details; in the second one, we will explain the changes applied on the CPC to implement anomaly detection; and in the last one, we will describe the encoder models chosen to be compared.

3.1. Contrastive Predictive Coding

Contrastive Predictive Coding (CPC) learns useful features by training neural networks in order to predict the representations of future observations from those of the past. In our application of this technique to the images domain, CPC works by predicting the part of the image below a certain position. In particular, the images are

divided into a grid of patches, and the goal of network, which will be able to see only the top part of the image, is to predict the remaining section. We could in fact interpret each row of patches of the image as a different time step, going from top to bottom, so the network, predicting the bottom rows, will hence model the "future" representations. The predictions are then calculated employing a contrastive loss, which has been developed in order to avoid the network from making trivial solutions such as representing the predictions part as a constant vector, which would be the case of mean squared error loss.
This loss is called InfoNCE, and the main idea behind it is that the network must correctly classify "future" representations among a set of unrelated ones.
In order to build CPC, the input image is divided into a grid of overlapping patches $x_{i,j}$ where i, j denote the position of the patch in the image. Each patch is then given as input to the encoder g_{enc} which transforms each patch into more dense and with lower spatial dimension representations, which is each patch becomes a vector, $z_{i,j} = g_{enc}(x_{i,j})$. Hence, the image becomes a grid of feature vectors.
A masked convolutional neural network is then used to make predictions. The mask are such that the receptive field of each resulting *context vector* $c_{i,j}$ is related only to feature vectors that rely above it in the image, which is $c_{i,j} = g_{ar}(\{z_{u,v}\}_{u \leq i,v})$, as shown in Figure 4.
The network makes prediction "future" vectors each $z_{i+k,j}$ from the context vectors $c_{i,j}$ where $k > 0$ and a prediction matrix W_k where the predicted feature vector is $\hat{z}_{i+k,j} = W_k c_{i,j}$ The predictions are then evaluated through a contrastive loss whose goal is to correctly recognize the target $z_{i+k,j}$ choosing between a set samples of random feature vectors z_l taken from the dataset. This loss, as defined in Oord et al. 2018 becomes:

$$\mathcal{L}_{CPC} = -\sum_{i,j,k} \log p(z_{i+k,j} \mid \hat{z}_{i+k,j}, \{z_l\})$$

$$= -\sum_{i,j,k} \log \frac{\exp(\hat{z}^T_{i+k,j} z_{i+k,j})}{\exp(\hat{z}^T_{i+k,j} z_{i+k,j}) + \sum_l \exp(\hat{z}^T_{i+k,j} z_l)}$$

This loss is inspired by Noise Contrastive Estimation [6] and has been shown to maximize the mutual information between $c_{i,j}$ and $z_{i+k,j}$ [11].

3.2. CPC for Anomaly Detection

Although the CPC aforementioned framework is widely exploited, some modification have been done in order to make this implementation more effective and precise.
First, since random samples are needed in order to compare them to the predicted ones, only negative samples (without anomalies) are drawn from the dataset in the testing phase.

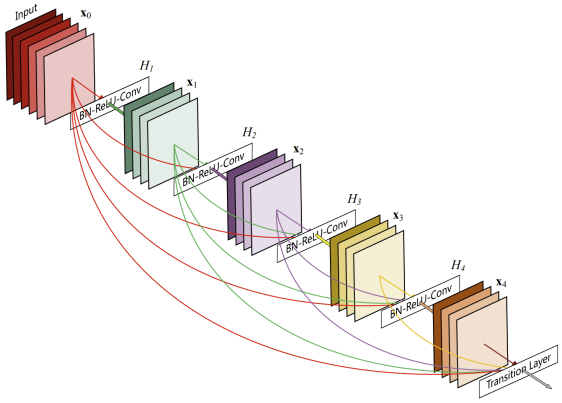

Figure 2. Connections between Densenet layers

This is done in order to prevent a predicted sample, whose goal is to reproduce a non-anomalous element, to be compared with an anomalous one.
Previous implementations of the CPC model use random patches from within the same test-batch [9] as negative samples. However, this may result in negative samples containing anomalous patches, which could make it harder for the model to detect anomalous patches in the positive sample based on the contrastive loss.
Second difference is that the autoregressive part of the model is omitted, making the model simpler but still able to learn useful latent representations. As a result, the predicted feature vector changes from $\hat{z}_{i+k,j} = W_k c_{i,j}$ to $\hat{z}_{i+k,j} = W_k z_{i,j}$ in our loss function. Figure 3 shows an overview of this method.
The loss calculated for each image is then compared with an implicit threshold value (since we use the area under the receiver operating characteristic urve (AUROC) as performance measure) in order to classify the image as anomaly or not.

3.3. Encoder models

We tested three different encoder models, two of them based on the well-known ResNet while the third was based on DenseNet. In particular, we first implemented the ResNet18-v2, in order to reproduce the model from "Contrastive Predictive Coding for Anomaly Detection", until the third residual block. However, we wanted to try out also the ResNet50-v2 in order to see if there were some differences in terms of results, since the residual blocks are built in a different way, again until the third residual block.
A completely new and promising architecture has been chosen: DenseNet. Whereas traditional convolutional networks with L layers have L connections, one between each layer, DenseNet has $\frac{L(L-1)}{2}$ direct connections, which is every convolutional layer has as input all the feature-maps

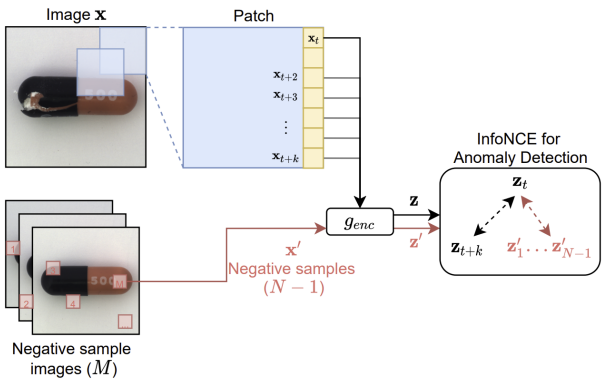

Figure 3. Overview of Contrastive Predictive Coding for Anomaly Detection

from the previous layers, and is connected to every other convolutional layer, as figure 2 shows.

This connections are made for many reasons, one of which is to ensure maximum information flow between layers (with matching feature-map sizes) directly with each other. In addition, this architecture also helps reducing the vanishing gradient problem. A counter-intuitive characteristic is that DenseNet requires fewer parameters than traditional convolutional networks because there is no need to re-learn redundant feature-maps. In order to understand why, we should consider traditional convolutional networks as algorithms with a state, that is passed forward from each layer to the next one. So each layer reads the state of the network from the previous layer, modifies it, and passes it forward. ResNet [8] in order to preserve this information already learned throughout the network, use skip connections, in particular identity transformations. Variations of ResNet, though, have been shown that many layers can be randomly be turned off and dropped during training [5]. This fact makes the ResNet similar to traditional networks but with many more parameters, because each layer has its own weights.

DenseNet, instead, explicitly differentiates between information that is added from information that needs to be preserved, adding a small set of feature-maps at each layer. Also, has been noticed that dense connections have a regularizing effect, which reduces overfitting on tasks with smaller training set sizes. Traditional networks use the output of l^{th} layer as input to the $(l+1)^{th}$ layer [1] which gives the transition $x_l = H_l(x_{l-1})$ where H_l can be considered as a non linear transformation.

ResNet, instead, add a skip connections that bypasses the non-linear transformation, hence: $x_l = H_l(x_{l-1}) + x_{l-1}$. This gives the advantage that the gradient flows through the network directly with these identity connections. Although this network architecture has reached incredible results, the summation between the identity function and the output of H_l could lead to a hard information flow in the network.

This, instead, does not happen in Densenet because each layer receive the feature-maps of all preceding payers, and the transition of each layer is: $x_l = H_l(x_0, x_1, ..., x_{l-1})$ where $(x_0, x_1, ..., x_{l-1})$ is the concatenation of the feature maps of each preceding layer.

This concatenation is not feasible when the size of the feature-maps changes, and to solve this some $1x1$ convolutions are added to fix the dimensions. In order to make everything work the network is divided into dense-blocks, where are present convolutions only, and between them there are transition layers, which make a $1x1$ convolution, in order to reduce the number of input feature-maps to the next layer [8] [2] and pooling, as we can see in figure 5.

Also, it was proved that this architecture is able to learn even with really limited number of filters, for example $k = 12$, which is also called growth rate of the network. This means that every layer only addes $k = 12$ feature maps to the "collective knowledge" of the network. However, for the implementation of our model, we only employed one dense block, preceded by a transition block, and followed by a convolutional layer and a max-pooling and a fully-connected layer in order to have the requested dimensions as output.

4. Experiments

Datasets. We evaluate the proposed anomaly detection model on the MVTec AD dataset. The mentioned dataset contains high-resolution images of ten object and five textures with pixel-accurate annotations and provides between 60 and 391 training images per class. We focus on the first five classes of this dataset (Bottle, Cable, Capsule, Carpet, Grid) due to computational time and power limitations.

The input images are resized to $256x256$ and divided to form a grid of $7x7$ patches of size $64x64$ with a 32 pixel overlap between each other.

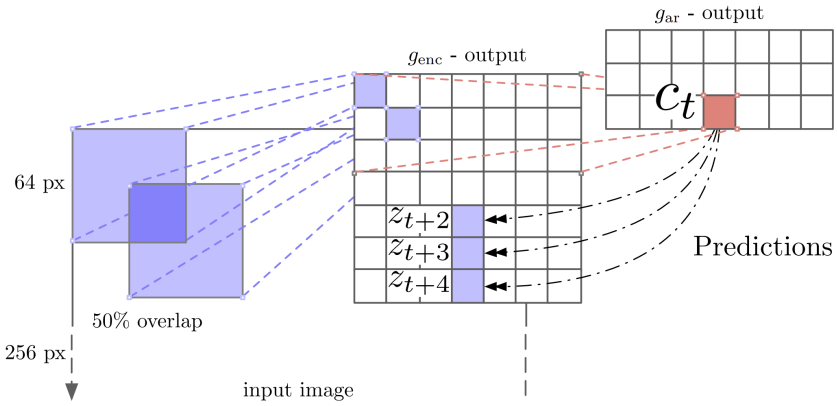

Figure 4. Contrastive Predictive Coding on image

Experiments setup. Each patch is given as input to the encoder, as first ResNet18-v2, whose outputs from its third residual block are subject to mean-pool in order to get a single 1024-d vector for each patch. The whole image will then have a $7x7x1024$ shape, as Figure 4 shows.
We trained a different model from scratch for each class for 30 epochs, and for Bottle and Cable we were able to also train for 55 and 70 epochs. Also, as described in the paper [7], in order to increase the accuracy of InfoNCE loss as indicator of anomaly images, we apply four separate models each one analyzing the image from a different direction and combining their loss in the test phase. This is done in order to increase the accuracy, because it could happen that analyzing an image in only one direction (for example from top to bottom) could hide part of an anomaly to the network. Every model is trained using Adam optimizer and learning rate $1e-5$ and batch size of 16. For both training and evaluation we use 16 non anomalous samples in the InfoNCE loss. The same choices are also applied to the models with ResNet-50 v2 and DenseNet as encoders architecture.
The Densenet model has been implemented using only one dense block, and with growth-rate (k) of 12. After the dense block a convolutional block followed by an average pooling layer have been added in order to have the required dimentions.

Results and discussion. Haan et al. (2021) works with images of size $768x768$, dividing each of them into patches of size $256x256$ with 50% overlap between them, hence having 25 patches of size $256x256$. These patches are divided in sub-patches of size $64x64$, again with 50% overlap between them, thus each patch will have 49 sub-patches. The whole image will so have $25 * 49 = 1.225$ of size $64x64$. We followed this procedure, but the training process was too slow for the gpu time allowed by the framework we used, which was Kaggle, so we decided to resize the images to $256x256$ and diving each in patches of size $64x64$. In this way we were able to train multiple classes and models. Consequently, the precision and performance dropped significantly.
To evaluate our model's performance for detecting anomalies, we average all the InfoNCE loss values of the patches within an image and use this value to calculate the AUROC score. In table 1, we compare different encoders on the same class and number of epochs. We find that the model with DenseNet as encoder has always better results with respect to the others and its performance generally increases as the number of training epochs increased. At the beginning, the AUC Grid class results were less than 0,5: something is gone wrong and we could not take into account these values. After a detailed search on the code, we find out that the value of the learning rate was different with respect to all the other class models and, after a new training with the correct hyperparameter value, l'AUC of this class increases given results greater than 0,5. There is no much difference between the performance of ResNet18 and ResNet50. The only big difference is in the Bottle class: ResNet50 outperforms ResNet18 especially if trained for many epochs.

5. Conclusion

We can ascertain that contrastive learning can be used for anomaly detection, which is a relatively new method. The proposed method could perform well on the anomaly detection task, especially with Densenet as encode architecture, using high-dimensional data. This algorithm could still be improved, one path to follow is to perform some fine-tuning on the parameters and modify some hyperparameters of the network. This contrastive predictive coding approach could be applied in order to make predictions in multiple other domains, that could space from time series, for example audio

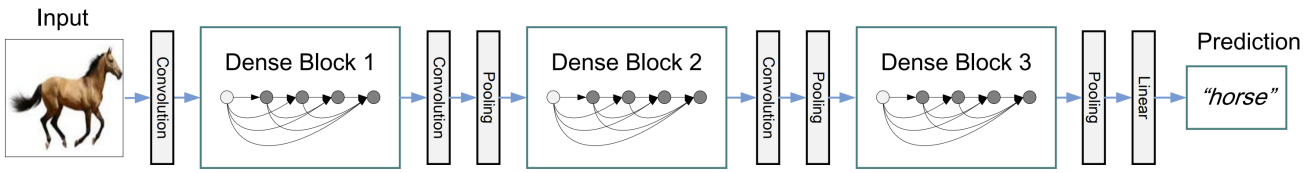

Figure 5. Overview of Densenet architecture

Class	Epochs	ResNet18	ResNet50	DenseNet
Bottle	30	0.558	0.650	0.790
	55	0.419	0.689	0.826
	70	0.449	0.704	0.876
Cable	30	0.770	0.644	0.840
	55	0.721	0.635	0.821
	70	0.736	0.651	0.858
Capsule	30	0.553	0.598	0.609
Carpet	30	0.565	0.597	0.618
Grid	30	0.547	0.581	0.612

Table 1. Anomaly detection AUROC score on the first five classes of MVTec-AD

signals to natural language processing.
It would be very interesting to test our algorithm on other datasets, and exploiting it not only for anomaly detection but also, for example, to predict images of multiple kinds of objects.

References

[1] I. S. A. Krizhevsky and G. E. Hinton. Imagenet classification with deep convolutional neural networks, 2012. 4

[2] S. I. J. S. C. Szegedy, V. Vanhoucke and Z. Wojna. Rethinking the inception architecture for computer vision, 2016. 4

[3] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41:15:1–15:58, 2009. 2

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations, 2020. 1, 2

[5] Z. L. D. S. G. Huang, Y. Sun and K. Q. Weinberger. Deep networks with stochastic depth, 2016. 4

[6] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. 3

[7] P. d. Haan and S. Löwe. Contrastive predictive coding for anomaly detection, 2021. 1, 5

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2016. 2, 4

[9] S. A. F. J. D. R. A. D. C. E. S. M. A. Henaff, O. J. and A. van den Oord. Data-efficient image recognition with contrastive predictive coding, 2020. 1, 3

[10] V. D. M. W. Huang, Liu. Densely connected convolutional networks, 2021. 1, 2

[11] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding, 2019. 1, 3

[12] C. Qiu, T. Pfrommer, M. Kloft, S. Mandt, and M. Rudolph. Neural transformation learning for deep anomaly detection beyond images. 2021. 2

[13] R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks, 2015. 2

[14] J. Tack, S. Mo, J. Jeong, and J. Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances, 2020. 2

[15] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks, 2016. 2

[16] S. Zagoruyko and N. Komodakis. Wide residual networks, 2016. 2