Anomaly Detection in Imbalanced Problems

- Deep Learning Project -

Davide Esposito Pelella, 1886977

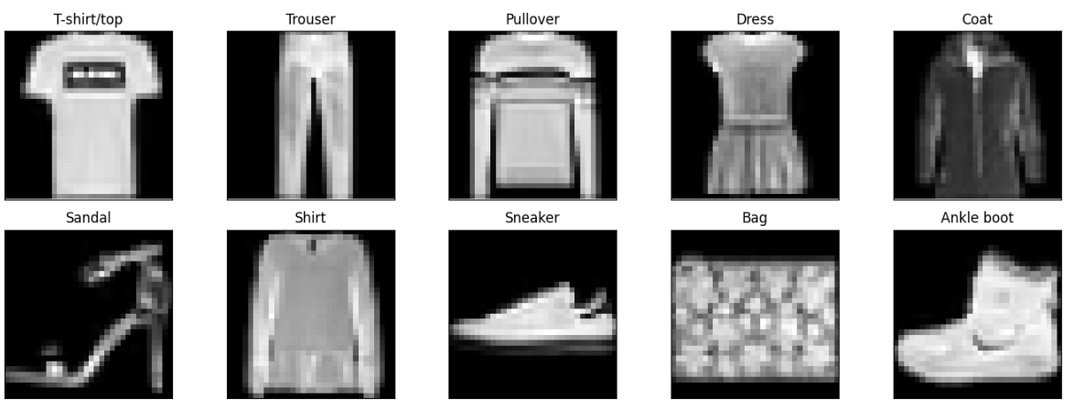
Introduction

1.1 Project Overview

The report which follows is the result of the project of the course of Deep Learning. The aim of this work was focused on the Anomaly Detection in Imbalance Problems, dealing with gap of samples and classes distributions, based on the paper: [1] *GAN-based Anomaly Detection in Imbalance Problems*. In particular this work implement the SOTA architecture explored in the paper's research and 2 more baseline to compare the results.

1.2 Dataset

The Dataset used is the FashionMNIST [2] , a dataset of Zalando's article images of 10 classes, with a total of 60,000 train samples and 10,000 test samples. Each sample is a 28x28 grayscale images (784 px) containing only 1 class. The dataset in particular is balanced, in training and test set.



Dataset

1.3 Data Augmentation

For the training was applied some Data Augmentation techniques to the FashionMNIST dataset, in particular the images were vertically and horizontally flipped, randomly cropped 0~2 pixels from the boundary, randomly rotated of 90, 180, or 270 degrees and resized to 32x32 dimension.

1.4 Preprocessing and Unbalance

The dataset in the proposed implementation is loaded by using a specific realized class called *AnomalyDetectionDataModule*, which load the dataset and provide the training and test dataloaders. The DataModule is realized in a flexible way, allowing to perform a training/test also only on a subset of the available data in order to perform tests. Furthermore the model will need to deal with the imbalance problems typical of anomaly detection.

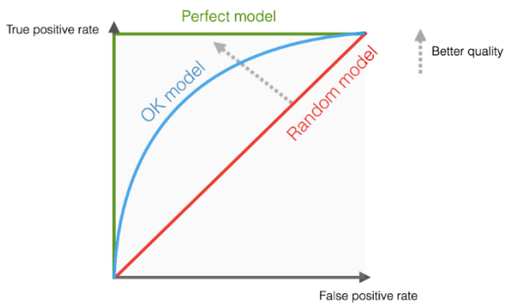
Class imbalance is a well known problem in Anomaly Detection, for which, usually, there is a class disparity between the normal and the anomal class. For this specific task with the FashionMNIST, each class is once considered as the 'normal class' at time, and the remaining 9 as 'anomal class', in this way, 10 experiments are performed and the resulting score is the mean among all. Due to samples availability this division lead

to an unbalanced situation and to deal with this problem was performed a *sampling* from each class recognized as 'anomal' in order to get the same number of the 2 classes, both now with 6000 samples in the total at training time.

Furthermore, if the data are sampled randomly, without considering their particular distribution, the resulting dataset won't be balanced in the distribution and representation of starting dataset. To solve this condition was applied a *k-means clustering* approach, in order to sample the same number of data within each cluster of each group, now then considering the distribution of the samples itself. This is obtained by using the *kmeans_sampling* function in the DataModule, which performs both the k-means clustering and sampling according to these requirements.

1.5 Metric

In order to evaluate the performance of the models on the dataset, the main metric used was the Area Under the Receiver-Operating Characteristics (ROCAUC) [3]. This metric, indicating the area under the ROC curve, sums up how well a model is able to produce relative scores to discriminate between positive or negative instances, which in our case is represented by the normal and anomalous samples of interest. The ROC curve is the plot of the true positive rate (TPR) against the false positive rate (FPR), at various threshold settings. It ranges from 0 (bad) to 1 (perfect), and 0.5 indicates the performance of random guessing.



ROC AUC score

1.6 Limitations

For the implementation was used *pytorch lightning* and all the train and test phases were performed by using the hardware resources provided by the free version of Google Colab. Unfortunately the free version of Google Colab has some limitation in computational power and resource disponibility. These limitations have negatively impacted the overall performances of the work reducing the possible training. The Google Colab policy indeed, after some hours of use, will progressively limit the resources provided to the user, not allowing to perform the same training showed in the paper.

Models

2.1 Baseline 1: Random Guessing

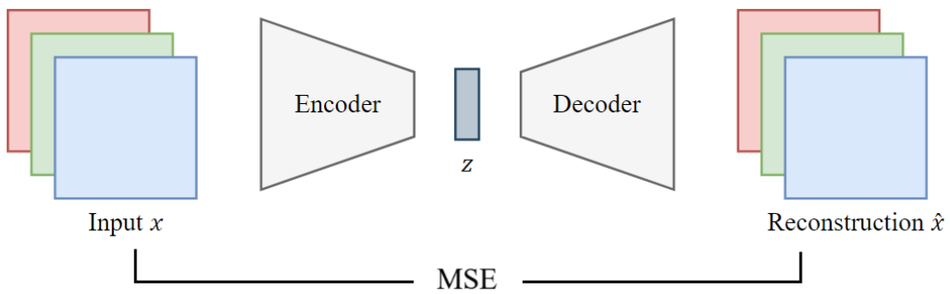
The first choosen model as baseline is a random guessing approach. The architecture is simple and straightforward, At each sample provided the model will randomly generate an output in the range [0,1], randomly guessing between the *normal* and *anomal class*.

Due to it's intrinsic nature, this approach didn't need any training, ad can be directly performed the test on the test-set. As expected the resulting AUROC computed was of 0.5 by considering a threshold of 0.5 .

2.2 Baseline 2: Autoencoder

The Autoencoder is often a popular choice for anomaly detection tasks. It consists of 2 main parts:

- *Encoder* = Map the input in a lower-dimensional latent space representation, in this case it consists of 2 convolutional layers with ReLU and MaxPooling. For the output activation function was choosen the sigmoid because this provided the best results for the model performances balancing the correctly reconstruction of 'normal class' and the worse for 'anomal classes';
- *Decoder* = Map from the lower-dimensional latent space trying to reconstruct the original data with 2 transpose convolutions.



For the training was only provided the *'normal class'* each time, and trained the model by trying to reduce the reconstruction error by minimizing the MSE (Mean Square Error) between input and reconstructed output.
The model was trained for 15 epochs, with *lr = 0.0001* with the Adam optimizer, and *batch_size = 1* to have comparable results with the sota architecture. This leaded to a resulting AUROC = 0.77 , averaged among the classes, with a threshold value of 0.5. Unfortunately the model often reconstruct sufficiently correctly also the anomal data, resulting in a lower AUROC performance.

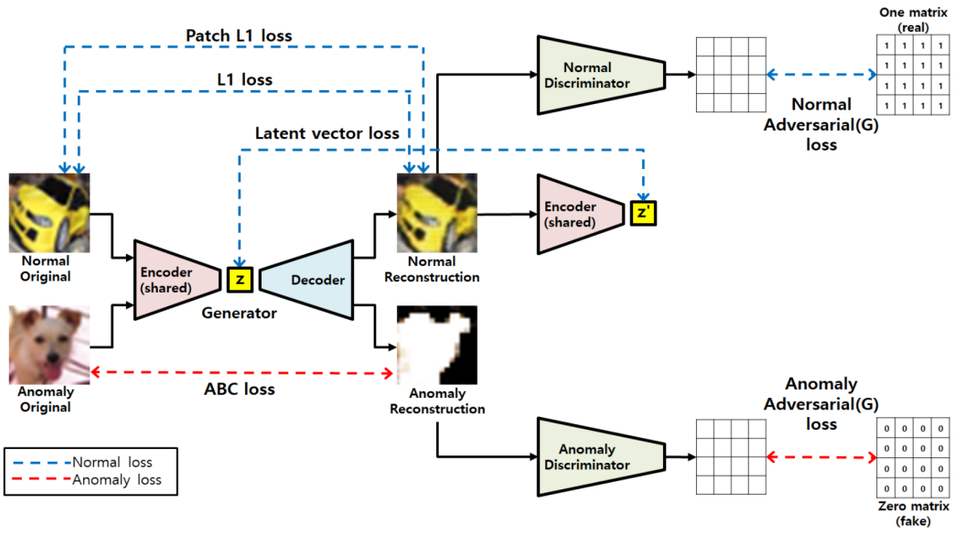
2.3 Baseline 3: GAN-based

This represent the GAN-based anomaly detection model, the state-of-art for the Benchmark datasets for the AUROC(%) presented in the paper. The network architecture is based on a GAN structure, consisting of a generator and a discriminator.

- *Generator* = It is a modified U-Net [4] structure, in the form of an autoencoder. It consists of four *4x4* convolutions with stride 2, then followed by 4 transposed *4x4* transposed convolutions with skip connections. For the activation function of output where tested many alternatives, but the tanh resulted to be the best suitable trade-off in the reconstruction of the images;
- *Discriminator* = The discriminator is a convolutional network, consisting of three *4x4* convolutions with stride 2 then followed by two *4x4* convolutions with stride 1.

The structure resulting was a slightly different model from the sota presented in the paper in terms of number of parameters. This because, even after many manual combinations, apparently is impossible to perfectly reproduce the number of parameters provided in the paper, then, to get the final architecture the structure was entirely based on the layers description provided and on the U-Net [4].

During the training phase 2 Discriminator are used to separately process the input data when the class was the normal or the anomal. The Generator is trained to output 1 from normal data and 0 from anomal one. In this way the model will learn to minimize reconstruction error when normal data is provided to Generator, while will try to maximize the reconstruction error when input is anomal.



As in the paper, for the training where used 6 types of loss functions to train the generator and 2 for the discriminators (1 for each), used and combined with a weighted summation as in the research mentioned. The double discriminator allow to solve the discriminator distributional bias, because the generator will be trained to output 1 from normal data and 0 from anomaly, while, if it was trained with only 1 discriminator, the model will be trained to only classify well normal images.

The model was trained for 7 epochs with Adam optimizer, a batch size of 1, a learning rate of 0,0001 , getting AUROC = 0.9199 averaged among the classes with a threshold of 0.5. Unfortunately the training was also heavly influenced by the weights initialization, which needed some trials in the execution of the single training to get better results.

Conclusions

In the following table are summarized the the performances of the models explored so far in the report.

Model	**AUROC**
Random Guessing	0.5
Autoencoder	0.7714
GAN	0.9199

Table 3.1: Comparison of Models

- The Random Guessing is by far the worst model among the three, representing the minimum baseline performances expected according to the AUROC metric. The testset is balanced, so in this case the result is quite perfectly addressed to a value of 0.5, but, due to the architecture this value is heavly influenced by the distribution of the samples.

- The Autoencoder on the other hand resulted in a really fast training, in particular if compared with the GAN, and lighter model. Due to the intrinsic nature of the approach, the model will always try to reconstruct in output the image provided as input, even the anomaly one is provided, so the performances are linked to this reconstruction error which sometimes give good results even with an anomal sample, sufficiently to lead to a miss-classification between normal and anomal.

- The GAN, in the end, resulted in a really slower training, due to the presence of 3 structures, but as expected, this approach provided the highest perfromances. This result is linked to the architecture and approach of the model which, not only try to correctly reconstruct a 'normal sample', but when provided an anomal

samples, will try to badly reconstruct on porpouse, limiting the mis-classifications between normal and anomal, due to the reconstruction error computed like could happen in the Autoencoder counterpart.

Bibliography

[1] Junbong Kim, Kwanghee Jeong, Hyomin Choi, and Kisung Seo (2020). *GAN-based Anomaly Detection in Imbalance Problems.*

[2] FashionMNIST dataset. *https://github.com/zalandoresearch/fashion-mnist.*

[3] Area Under the Receiver-Operating Characteristics. *https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc$_a$uc$_s$core.html*.

[4] Ronneberger, Fischer, Brox. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)*https://arxiv.org/abs/1505.04597.*