Anomaly Detection with Robust Deep Autoencoders

original article by C. Zhou and R. C. Peffenroth

Alessandro Trenta

Scuola Normale Superiore

Deep Autoencoders

- A Deep Autoencoder (DAE) is constituted by two main components: an encoder E and a Decoder D.
- The main objective of a DAE is to learn the identity map so that the reconstruction $\bar{X} = D(E(X))$ is as close as possible to the original input X.
- The encoder and decoder functions E, D can be any kind of mapping between the data space and the coded space. Usually they are Deep Neural Networks e.g. FCNN complex models such as Long Short Term Memory (LSTM).
- The objective is usually to find the minimum reconstruction error w.r.t. some parametrized encoding and decoding functions and a distance (in this case the L_2 norm)

$$\min_{\theta,\phi} \|X - D_\theta(E_\phi(X))\|_2 \tag{1}$$

Principal Component Analysis

- Assume to have a set of N samples of n dimensional data, so that $X \in \mathbb{R}^{N \times n}$ s.t. each column has 0 mean (we can just shift the data to fulfill this request).
- Principal Component Analysis (PCA) is defined as an orthogonal linear transformation such that the new coordinate system of $\mathbb{R}^n$ satisfies: the i-th component of the coordinate system has the i-th greatest data variance if we project all samples on that component.
- Ideally we are trying to fit a n-ellipsoid into the data. The length of an axis of the ellipsoid represents the variance of data along that axis.
- PCA is often used for dimensionality reduction or encoding: we can project the data on the first $k < n$ principal components.

Principal Component Analysis

Mathematically we can define:

$$w_1 = \underset{\|w\|_2=1}{\arg\max} \|Xw\|_2^2 = \underset{w}{\arg\max} \frac{w^T X^T X w}{w^T w} \tag{2}$$

for the first component. Then for the k-th component we first subtract the first $k-1$ principal component from X

$$\hat{X}_k = X - \sum_{i=1}^{k-1} X w_i w_i^T \tag{3}$$

and finally solving again the similar problem:

$$w_k = \underset{\|w\|_2=1}{\arg\max} \|\hat{X}_k w\|_2^2 = \underset{w}{\arg\max} \frac{w^T \hat{X}_k^T \hat{X}_k w}{w^T w} \tag{4}$$

Robust Principal Component Analysis

- Robust Principal Component Analysis (RPCA) is a generalization of PCA that aims to reduce the sensitivity of PCA to outliers.
- The idea is to find a low-dimensional representation of data cleaned from the sparse outliers that can disturb the PCA process.
- We therefore assume that data X can be represented as $X = L + S$: L has low rank and is the low-dimensional representation of X while S is a sparse matrix consisting of the outlier elements that cannot be captured by the representation.

Robust Principal Component Analysis

- The problem can be addressed as:

$$\min_{L,S} \rho(L) + \lambda \|S\|_0 \tag{5}$$

$$\text{s. t. } \|X - L - S\|_F^2 = 0 \tag{6}$$

  where $\rho(\cdot)$ is the rank of a matrix and we used the zero norm.
- This optimization problem is NP-hard and tractable only for small metrices.
- Usually it is substituted by the following problem, which is convex and tractable also for large matrices:

$$\min_{L,S} \|L\|_* + \lambda \|S\|_1 \tag{7}$$

$$\text{s. t. } \|X - L - S\|_F^2 = 0 \tag{8}$$

  where $\|\cdot\|_*$ is the nuclear norm i. e. the sum of singular values of a matrix.

Robust Deep Autoencoders

- The main idea behind Robust Deep Autoencoders (RDAE) is to combine the representation learning of DAEs and the anomaly detection capability of RPCA.
- Noise and outliers are incompressible in the lower dimensional space we want to represent our data in.
- The objective is to learn a good low dimensional representation except for few exceptions.
- We will see two RDAE typed, one for l_1 regularization and one for $l_{2,1}$.

RDAE with l_1 regularization

- The RDAE objective is to decompose data $X = L_D + S$ just as in RPCA.
- By removing the noise S the autoencoder can better reconstruct L_D.
- As before, the best choice to obtain a sparse S would be to use a loss of the type $\|S\|_0$ which counts the non-zero entries, solving the problem

$$\min_{\theta} \|L_D - D_\theta(E_\theta(L_D))\|_2 + \lambda \|S\|_0 \tag{9}$$

$$\text{s.t. } X - L_D - S = 0 \tag{10}$$

- The parameter λ controls the sparsity of S and plays an essential role.

The role of λ

- As we said, the role of λ is very important.
- A smaller λ means that the norm of S plays a less important role and much of the loss will come from the DAE.
- The model will reconstruct better but recognize less outliers. This could be helpful if we want a more faithful representation e.g. for supervised tasks.
- A larger λ, instead, gives more importance to the norm of S as a loss.
- This means that the model will recognize more (or even too much) outliers, sacrificing some reconstruction performance.

The true objective

- As for the RPCA the previous loss is non tractable. We then instead focus on the following problem:

$$\min_{\theta} \|L_D - D_\theta(E_\theta(L_D))\|_2 + \lambda \|S\|_1 \tag{11}$$

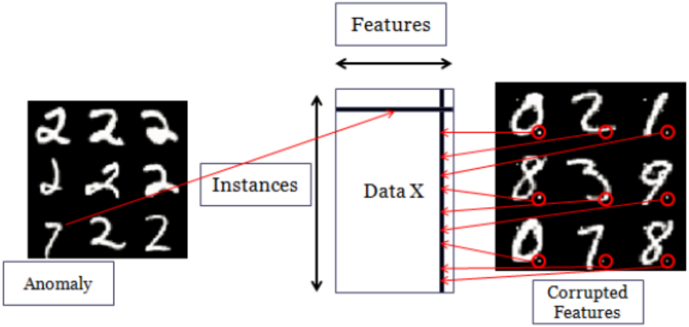
$$\text{s.t. } X - L_D - S = 0 \tag{12}$$

Notice two things:

- The autoencoder is trained with L_D, the part of its decomposition assumed to be outliers and noise free.
- There is no specific requirement about the DAE, in fact D_θ and E_θ are generic decoder, encoder functions.

RDAE with $l_{2,1}$ regularization

- The RDAE with l_1 penalization assumes that outliers and noise are not structured. The l_1 penalty is indeed just a regularization to induce sparsity.
- In general we can have multiple types of errors: an instrument that corrupts an input feature or outliers where the input data is in some way structurally different from normal data.


Features
Instances
Data X
Anomaly
Corrupted Features

The $l_{2,1}$ norm

- The $l_{2,1}$ norm is defined as ($X \in \mathbb{R}^{N \times n}$):

$$\|X\|_{2,1} = \sum_{j=1}^{n} \|X_j\|_2 = \sum_{j=1}^{n} \left(\sum_{i=1}^{N} |X_{ij}|^2 \right)^{\frac{1}{2}} \tag{13}$$

- The $l_{2,1}$ norm can be seen as introducting a l_2 norm regularization over each feature and then adding a l_1 regularization accross features.
- We can also do the other way around: to recognize data anomalies (by row) just apply the $l_{2,1}$ norm to X^T.

- The final optimization problem for the RDAE with $l_{2,1}$ regularization for data anomalies is then

$$\min_{\theta} \|L_D - D_\theta(E_\theta(L_D))\|_2 + \lambda \|S^T\|_{2,1} \tag{14}$$
$$\text{s.t. } X - L_D - S = 0 \tag{15}$$

- For detecting feature anomalies we just need to change the objective to

$$\min_{\theta} \|L_D - D_\theta(E_\theta(L_D))\|_2 + \lambda \|S\|_{2,1} \tag{16}$$
$$\text{s.t. } X - L_D - S = 0 \tag{17}$$

The proximal operator

- To see in detail the training procedure for the RDAE we first need to consider the proximal operator.
- For general optimization problems of the form $\min f(x) + \lambda g(x)$ where g is convex some of the most used methods require to find

$$\text{prox}_{\lambda,g}(x) = \arg\min_y g(y) + \frac{1}{2\lambda}\|x-y\|_2^2 \tag{18}$$

- In this case we then want to obtain a solution of the problems

$$\text{prox}_{\lambda,l_1}(x) = \arg\min_y l_1(y) + \frac{1}{2\lambda}\|x-y\|_2^2 \tag{19}$$

$$\text{prox}_{\lambda,l_{2,1}}(x) = \arg\min_y l_{2,1}(y) + \frac{1}{2\lambda}\|x-y\|_2^2 \tag{20}$$

- For the l_1 norm, the solution to the proximal problem is

$$\text{prox}_{\lambda,l_1}(x) = \begin{cases} x_i - \lambda, & x_i > \lambda \\ x_i + \lambda, & x_i < -\lambda \\ 0, & x_i \in [-\lambda, \lambda] \end{cases} \tag{21}$$

  which in the case of $S \in \mathbb{R}^{N \times n}$ gets applied element by element.
- For the $l_{2,1}$ norm, we obtain (let $S_{\cdot j}$ be the column vector $S_{ij}, j = 1, \dots, N$)

$$(\text{prox}_{\lambda,l_{2,1}}(S))_{ij} = \begin{cases} S_{ij} - S_{ij} - \lambda \frac{S_{ij}}{\|S_{\cdot j}\|_2}, & \|S_{\cdot j}\|_2 > \lambda \\ 0, & \|S_{\cdot j}\|_2 \leq \lambda \end{cases} \tag{22}$$

  if we are considering feature wise anomalies, substitute S with S^T for data anomalies.

The main algorithm

- The method used to train the RDAE is the Alternating Direction Method of Multipliers (ADMM).
- The main idea is to optimize the problem

$$\min_{\theta} \|L_D - D_\theta(E_\theta(L_D))\|_2 + \lambda \|S^T\|_{2,1} \tag{23}$$

$$\text{s.t. } X - L_D - S = 0 \tag{24}$$

  by doing it in two steps at each iteration.
- First, we fix S and optimize the DAE loss $\|L_D - D_\theta(E_\theta(L_D))\|_2$ with backpropagation as usual.
- Then, we fix L_D and optimize the regularization term with the proximal method.

The full procedure is the following: given input $X \in \mathbb{R}^{N \times n}$, initialize $L_D \in \mathbb{R}^{N \times n}$, $S \in \mathbb{R}^{N \times n}$ as zero matrices, $L_S = X$ and initialize the DAE randomly. For each iteration do:

- $L_D = X - S$
- Minimize $\|L_D - D_\theta(E_\theta(L_D))\|_2$ with backpropagation.
- Set $L_D = D(E(L_D))$ as the reconstruction.
- Set $S = X - L_D$.
- Optimize S using a $\text{prox}_{\lambda, l.}$ function of choice.
- If $c_1 = \frac{\|X - L_D - S\|_2}{\|X\|_2} < \epsilon$ or $sc_2 = \frac{\|LS - L_D - S\|_2}{\|X\|_2} < \epsilon$ we have early convergence.
- Set $L_S = L_D + S$.

Return L_D and S.

Results

- I tried to reproduce some of the results by the original article.
- For the main article results the database used was the MNIST digits database.
- The train data contains 50000 samples while the test set contins 10000 images.
- Data was flattened from images of shape $(28, 28, 1)$ into vectors of length 784. Train data is then a matrix in $\mathbb{R}^{50000 \times 784}$.
- Pixel walues are converted from integers between 0 and 255 to floats between 0 and 1.

Implementation

- The RDAE and the standard DAEs used in this experimental tries were implemented using Tensorflow 2.9.1 on python 3.8.
- For the random forest classifier and the isolation forest models were taken from SciKit-learn version 1.1.1.
- Capire se mettere implementazione o no

l_1 Robust Deep Autoencoder

- To assess the performance of the l_1 RDAE the proposed procedure is the following:
- The training images get corrupted with a percentage of pixel (from 5% to 50%) changed to a random value between 0 and 1.
- Both the RDAE with l_1 regularization and a standard DAE (with same architecture as the DAE from the RDAE) are trained on these corrupted images.
- A random forest classifier is then trained on the feature extracted at the bottomneck layers of the two models.
- We test how these RF classifiers perform on the test set.

- This let us see how well the model is able to extract the important features of the images in a meaningful way.
- In this case the RDAE and DAE need to denoise the images and recognize which charactheristics are improtant.
- Both architectures are simple FCNN with layers of size 784 (input), 200 and 10 (the bottleneck and hidden feature layer).
- specificare parametri finali

INSERIRE RISULTATI E COMMENTI

$l_{2,1}$ Robust Deep Autoencoder

- The anomaly detection experiment is based on the recognizing of a specific digit of the MNIST dataset.
- First all the 4 digit images in the training set are collected in our dataset.
- Then, some images are chose at random from all the other digits until they are around 5% of total images in the dataset.
- This will be considered as the outliers of our data.

- The $l_{2,1}$ RDAE is trained on this dataset without any side information.
- Without telling the model which images are 4 digits and which are outliers the model itself must recognize the latters from original data on his own.
- The model architecture is the same as for the l_1 RDAE experiment.
- The only parameter that is fine tuned is the

- Model performance is assessed by seeing how it is able to recognize the correct "outliers".
- The metrics used are the accuracy, the precision score, the recall score and the F1 score defined down below.

$$ACC = \frac{TP + TN}{P + N} \quad P = \frac{TP}{TP + FP} \tag{25}$$

$$R = \frac{TP}{TP + FN} \quad F1 = 2\frac{P \cdot R}{P + R} \tag{26}$$

- The F1 score, which tries to average in some way precision and recall, is the metrics used to select the λ parameter.
- λ is the only parameter that is fine tuned (in a semi-supervised way).

- The performance of the RDAE as outlied detector is compared with the one obtained using the isolation forest method.
- The isolation forest method was a SOTA method for outlied detection. It is based on the idea that outliers are few and different and are separated from the rest.
- These outliers gets recognized using isolation trees which try to separate points from others.
- The only parameter which is peculiar to the method and which gets selected with the same metrics is the outlier fraction.

INSERIRE COMMENTI E RISULTATI + EVENTUALE PARTE SU LSTM https://github.com/AlexThirty/SaMLMfTSA

Thank you!

Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha.
Unsupervised real-time anomaly detection for streaming data.
Neurocomputing, 262:134–147, 2017.
Online Real-Time Learning Strategies for Data Streams.

Emmanuel J. Candes, Xiaodong Li, Yi Ma, and John Wright.
Robust principal component analysis?, 2009.

Neal Parikh.
Proximal algorithms.
Foundations and Trends in Optimization, 1:127–239, 01 2014.

Chong Zhou and Randy C. Paffenroth.
Anomaly detection with robust deep autoencoders.
Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, page 665–674, 2017.