Machine Learning & Data Mining Project - Airbus Challenge

Andrei Mardale Mohammad Poul-Doust Laetitia Couge Etienne Ekpo

Abstract

Detecting anomalies on timeseries dataset has become a very essential task in this era. Also, the rapid growth of sensor devices along with the data they generate raises further the urgency to design suitable machine learning algorithm not only to detect system/devices failure but predict as well this outcome before their occurrence.

1. INTRODUCTION

In this paper, we discuss various machine learning algorithms used in preprocessing and learning patterns on high dimensional sequences of Airbus helicopter flight generated data in view of detecting anomalies. This is a totally unsupervised setting in which only normal flight sequences have been availed in the training set. We will first of all review several papers that fall within this setting, provide an overview of the provided dataset and discuss our methodologies along with their respective results. It is worth noting that this project was initiated as a kaggle-like challenge organized by Airbus and INSA Toulouse.

2. RELATED WORK

The problem of anomaly detection is an important problem that has been widely studied in the literature. A fair amount of the existing work is using approaches based on cross-entropy loss which assumes the prediction that every input has to belong to one of the output class with a probability and naturally that is not very effective in case of anomaly detection.

In (Masana et al., 2018), a metric learning approach has been proposed to overcome the disadvantage of cross-entropy loss. A Siamese deep neural network has been adapted to train using a contrastive where points from same distribution are separated from other with same distribution but different class and also all in-distribution points are well-separated from out-of-distribution points. The method were evaluated against different benchmark datasets and results were "comparable or better" to the state of the art methods.

(Sani & Ghasemi, 2015) Tackled the problem of detecting any anomalous behavior in the network. The main contribution was to improve an unsupervised svm-clustering algorithm by using a custom learned distance function during the clustering. The original method clusrered data into k clusters and learned local SVM model for each cluster. The misclassified normal instances are passed to a metric learning algorithm. Due to similarity constraints, the false-positives with their nearest support vectors should be close (as misclassified point must be lying on the borders originally). Additionally, every data points should be as close as possible to its corresponding cluster's center. On the other hand, as dissimilarity constraints, every data point should be as far as possible from the centers of others' clusters. The method outperforms the original method on the same dataset.

Anomaly detection is usually accompanied with imbalanced data. and since standard metric learning algorithm tend to favor the majority class, (Gautheron et al., 2019) proposed an algorithm to learn from imbalanced data by learning an a distance tailored for imbalanced setting. This is achieved using standard metric learning constraints with additional formulation to treat each subset of similar (respectively dissimilar) pairs into two subsets for each class. Finally, a re-weighting strategy is applied to account for the imbalance in the data. The experiments for this method outperforms the state of the art metric learning algorithm in terms of F1-measure for imbalanced datasets and yet for the balanced ones.

In their work, (Cao V.L., 2016) discuss the use of both Autoencoder (AE) and Density based Models to detect anomalies in an unsupervised setting. After learning a good representation, anomalies can be identified by a threshold tuned on the reconstruction error(RE). Authors of this paper propose to use the embedded representation in the AE as input to train a Gaussian Density Estimator (KDE). Thus, the KDE learns the probability density of the normal data without the need to manually tune a threshold. The experimental results conducted on their benchmarks showed that the proposed hybrid model outperformed typical AE.

3. DATA DESCRIPTION

3.1. Input data

The Airbus dataset is built from flight data recorded by accelerometers. Each flight is sliced in sequences of 60 seconds; each sequence contains 61440 data points. Some

flights contain anomalous data, all the sequences extracted from these flights are flagged as anomalous. There exist different types of anomalies in the flights. We are provided with 3 datasets: the training dataset (1677 sequences) is a subset of normal samples, the validation dataset (594 sequences) and testing dataset (1917 sequences) contain normal and abnormal samples.

3.2. Data transformation

To reduce the dimension of the data, we have applied some transformations. Wavelet transform, contrary to Fourier Transform which uses frequency domain only, allows an analysis both in time and frequency, it can measure the evolution of frequency transients. We first apply a 5 point rolling average on the data trace. The Discrete Wavelet Transform (DWT) applied on the average time series produces series of coefficient from which we extract 120 features (statistics, entropy and zero crossings).

4. METHODOLOGY

The developed strategy for detecting the anomalies consists of two pipelines, one for training-validation set and one for test set. We present in this section the data flow in these pipelines from the input to the detection of the anomalies.

4.1. Training-Validation Pipeline

As the training set contains only normal points we focused on learning what correctness means. We build a set of classifiers aiming to learn the structure of the training set. However, some of the examples in the training set were further away from the others in terms of mean and variance, so we removed them first.

Figure 1. Validation Data with the True Labels after performing DWT feature extraction (Salagean & Firoiu, 2010) and a t-SNE for visualization. The blue points inside the cluster of red points represent difficult anomalies.

The analyse of the validation set showed us that some anomalies could be easily spotted by most of the classifiers while others anomalies which are clustering with normal point as shown in Figure 1 in the bottom left corner, seem to be more difficult to detect. This observation lead us to further split the Training-Validation Pipeline in two parts:

1. **Stage-1 Anomalies** - for detecting easy anomalies with a high confidence, focusing on a high precision.
2. **Stage-2 Anomalies** - for extracting and isolating "difficult" anomalies, focusing on a high recall.

4.1.1. Stage-1 Anomalies

We first target the easy anomalies, for which we propose the methodology described in Figure 2. This sub-part only uses the training set while the labels from the validation set are not required yet.

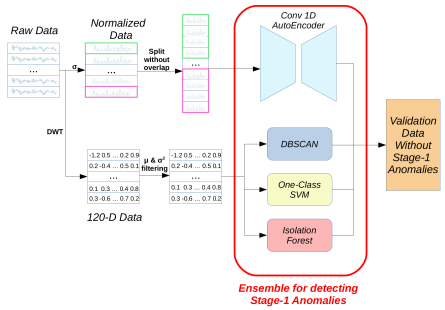

Figure 2. The first sub-part of the Training-Validation pipeline. It aims to detect anomalies with a high confidence and does not assume any risks regarding the more difficult ones.

The raw training data is processed by two different methods, after which the results of all the classifiers are ensembled to obtain a robust detection.

In the first method, raw time series are normalized and split into chunks of 384 points. The small chunks are fed to a Convolutional 1D Auto-Encoder. The normalization is performed applying a sigmoid function to bring the data in the range [0,1] as it was shown in (Vinayakumar et al., 2017) that the AE will perform better this way. When training the AE whose architecture is presented in Figure 3, we minimize the following reconstruction error:

$$MSE(X, \hat{X}) = \frac{1}{NP} \sum_{i=1}^{N} \sum_{j=1}^{P} (\hat{X_{i,j}} - X_{i,j})^2$$

where N is the total number of training examples (chunks), P is the length of the examples (384), X is the input data and $\hat{X}$ is the reconstructed data.

Then, the distribution of the reconstruction errors is plotted (4) and we fix a threshold for the loss beyond which the sample is treated as anomaly. By applying the Auto-Encoder on the validation set, we are able to flag some anomalies. We decided that a threshold value of 20 would be safe and not too detrimental.

In the second method we are using Representation Learning method. We are using as input the 120 extracted features

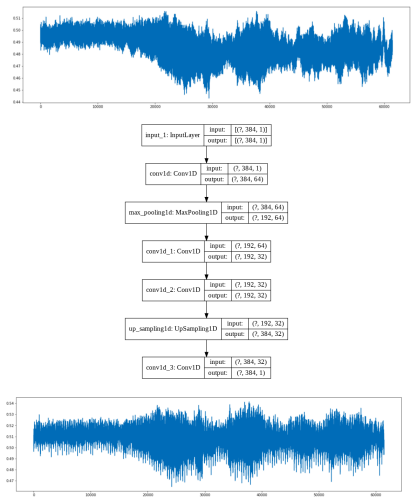

Figure 3. The architecture of the Convolutional Auto-Encoder

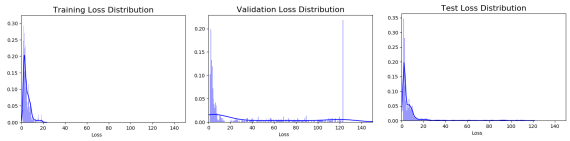

Figure 4. The loss distribution for the 3 data sets.

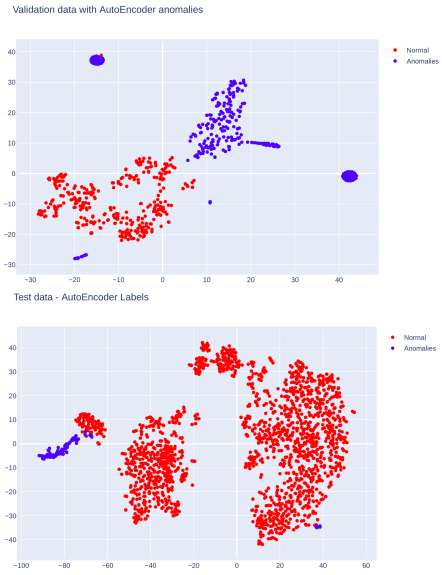

Figure 5. The validation and the test set labelled by the Stage-1 classifiers.

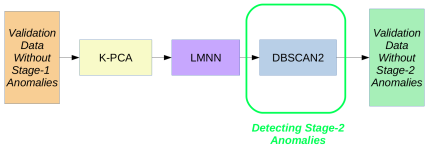

Figure 6. The second sub-part of the Train-Validation pipeline, aiming to detect more difficult anomalies, using Metric Learning.

described in the data Description section. We then apply a filtering strategy based on mean and variation of the new points and detect directly some of the anomalies. After the z-score normalization, we train a suite of 3 different classifiers and tune them all to detect anomalies only with a high confidence.

Finally, the end of the Training-Validation pipeline consists of ensembling the 4 different approaches and keeping the anomalies which are voted majoritarily. We can observe in Figure 5 the results on the Validation and on the Training set of the Stage-1 Anomaly detection process. As we aimed, the Stage-1 classifiers are not risking at all, thus we have a very low number of detected anomalies in the test set.

4.1.2. Stage-2 Anomalies

The second stage in the process aims to refine the result, by seeking harder anomalies, as the ones shown in the middle of the red cluster in Figure 1. We use an approach based on Metric Learning(ML), as presented in Figure 6.

Using ML, we aim to pull the normal examples, while the anomalies would be pushed away. Thus, ML suits perfectly our needs: we obtain implicitly another representation of the points, by learning a new pairwise distance between them, taking into account the semantic information, which is the labels we managed to collect for the validation data.

The first challenge we met was that the algorithm we use, Large Margin Nearest Neighbor (Weinberger & Saul, 2009), does not work with non-linear data. To overcome this issue, we applied a kernelized version of Principal Components Analysis (Mika et al., 1999), using a gaussian kernel to catch the non-linearities without reducing the dimension. Thus, we learn the metric in a new feature space induced by a non-linear kernel (Chatpatanasiri et al., 2010). Figure 7 shows the interesting effects of applying Metric Learning after k-PCA. We can observe how in the first picture, right after k-PCA some anomaly points were projected further away, but after LMNN, there is a clear separation between the two types of points.

We then learn a hyper-sphere around the normal points, using DBSCAN, as depicted in Figure 6, which will be also used for the test data-points.

4.2. Testing Pipeline

For the testing pipeline we use a similar flow as for the training-validation pipeline, which we describe in Figure 8.

After obtaining a preliminary set of anomalies, using the ensemble of Auto-Encoder, DBSCAN, One Class SVM and

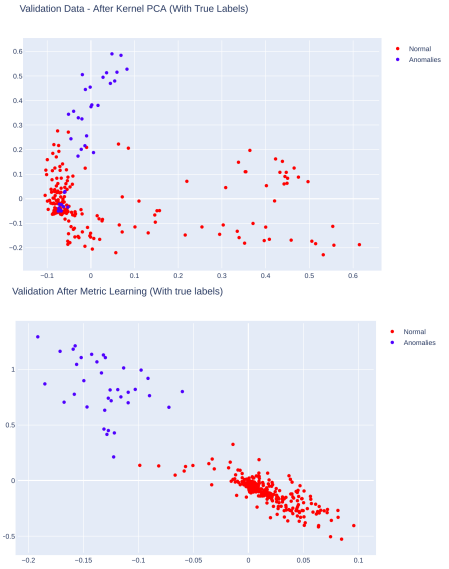

Figure 7. The Validation dataset, with true labels, after applying k-PCA, gaussian kernel, keeping 120 dimensions and the results after applying LMNN.

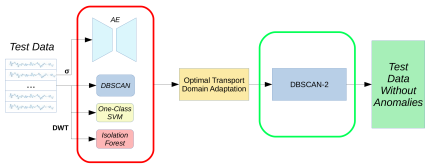

Figure 8. The testing pipeline. We run the data-points through the same 2 stages algorithm

Isolation forest, we remove them from the testing set and proceed with the next step, that is Optimal Transport. We are aware that the distribution of the testing set is totally shifted from the one of the validation set, so we tried to solve this issue using OTDA. Namely, we tried to use Sinkhorn's algorithm (Courty et al., 2016), (Rakotomamonjy et al., 2015) for Domain Adaptation OT with L1 and L2 regularization. However, the results we obtained were not satisfactory enough to give use confidence to use the method in any of our submissions. We ended up having points from the two classes being transported in the same vicinity, which we know from theory, that it should not happen. As a future work, we believe that this is definitely the key to obtaining much better results on the testing set.

After removing the easy anomalies, and applying k-PCA, we use the fitted LMNN model to project the testing points in the new implicit, induced space after computing the Mahalanobis distance. Then we use the DBSCAN-2 model that was trained on the validation set, to detect harder anomalies in the testing set. The results can be observed in Figure 9.

Finally, when we plug together the Stage-1 and Stage-2

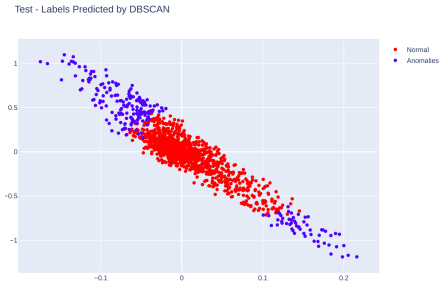

Figure 9. Testing set projected in the new space by the metric learning algorithm, with labels given by the DBSCAN model.

anomalies, we obtain the result depicted in Figure 10.

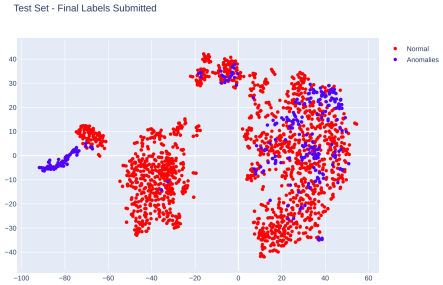

Figure 10. Testing set with all the labels that were given by Stage-1 and Stage-2 algorithms.

5. CONCLUSIONS

In this paper, we proposed different models for anomaly detection. Our approach consisted primarily on reducing the high-dimensionality of the data by extracting several time-series features upon which we applied in a first stage classic One class classification models such as AE, One Class SVM, Isolation forest and DBSAN in order to deal away with easy-to-detect anomalies in a totally unsupervised way. The second phase, which was semi-unsupervised, consisted in applying a non-linear transformation and learning a new distance metrics that bring together similar examples while pushing away dissimilar ones.

While our approach performed well on the validation set, it did not on the test set. That could be justified as both sets came from different distributions and hence, further improvement could involve the use Domain Adaptation techniques to obtain more accurate results.

References

Cao V.L., Nicolau M., M. J. A hybrid autoencoder and density estimation model for anomaly detection. *Parallel Problem Solving from Nature*, 9921, 2016.

Chatpatanasiri, R., Korsrilabutr, T., Tangchanachaianan, P.,

and Kijsirikul, B. A new kernelization framework for mahalanobis distance learning algorithms. *Neurocomputing*, 73(10-12):1570–1579, 2010.

Courty, N., Flamary, R., Tuia, D., and Rakotomamonjy, A. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9): 1853–1865, 2016.

Gautheron, L., Morvant, E., Habrard, A., and Sebban, M. Metric learning from imbalanced data. *arXiv preprint arXiv:1909.01651*, 2019.

Masana, M., Ruiz, I., Serrat, J., van de Weijer, J., and Lopez, A. M. Metric learning for novelty and anomaly detection. *arXiv preprint arXiv:1808.05492*, 2018.

Mika, S., Schölkopf, B., Smola, A. J., Müller, K.-R., Scholz, M., and Rätsch, G. Kernel pca and de-noising in feature spaces. In *Advances in neural information processing systems*, pp. 536–542, 1999.

Rakotomamonjy, A., Flamary, R., and Courty, N. Generalized conditional gradient: analysis of convergence and applications. *arXiv preprint arXiv:1510.06567*, 2015.

Salagean, M. and Firoiu, I. Anomaly detection of network traffic based on analytical discrete wavelet transform. In *2010 8th International Conference on Communications*, pp. 49–52. IEEE, 2010.

Sani, R. A. and Ghasemi, A. Learning a new distance metric to improve an svm-clustering based intrusion detection system. In *2015 The International Symposium on Artificial Intelligence and Signal Processing (AISP)*, pp. 284–289. IEEE, 2015.

Vinayakumar, R., Soman, K., and Poornachandran, P. Applying convolutional neural network for network intrusion detection. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1222–1228. IEEE, 2017.

Weinberger, K. Q. and Saul, L. K. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.