Mini-Project (ML for Time Series) - MVA 2024/2025

Guillaume Jarry guillaume.jarry@etu.minesparis.psl.eu
Mouad Id sougou mouadidsougou@gmail.com

April 20, 2025

1 Introduction and contributions

Anomaly detection in sequential data is a fundamental problem with applications in network security and industrial control systems. In particular, discrete sequences have unique properties related to their symbolic representation, hence it is an interesting topic to study. In this context, the paper we selected [1]. provides a comprehensive overview of existing techniques for detecting anomalies in symbolic sequences. Its main objective is to unify the theory of anomaly detection on symbolic sequences within a single framework.

We decided to implement most of the techniques detailed in section IV of the article to create a new python library, called **DAD** (**D**iscrete sequence **A**nomaly **D**etection). We then performed two experiments to test the implementation of our methods in python.

The first is a benchmark of the Markov Based techniques (section IV.3 of the article) on synthetic Markov chain which we generated ourselves. The second experiment was the application of a kernel based technique (section IV.1 of [1]) on continuous data that was transformed into symbolic sequence using the SAX algorithm [2].

Regarding the collaboration part, **Mouad** focused on implementing the Markov-based techniques, and deployed the Kernel based techniques on real data after applying SAX and studied the effect of discretization on the performance of the algorithm. **Guillaume** was heavly involved in the coding of the library, coding the window-based techniques and kernel-based techniques, as well as refactoring the Markov-based methods and benchmarking them on synthetic data. He also developed the utils module of the library, which includes the SAX transformation method, and created the dataset explorer notebook, documenting the exploration of datasets, including an analysis of the various dataset, which we did not document in this article since they failed.

The library is fully available on GitHub at this link. We coded everything from scratch, with some help from ChatGPT for the hardest algorithm, however it does not represent more than 30 percent of the overal code and its input were heavily modified (refactoring classes and fucntion for example).

2 Method

In this section, we will detail the functioning of the algorithm which we implemented in the DAD library. All these methods requires a training set of sequences that are considered as a baseline, ie "non anomalous". At testing time, the samples will be assessed against the model learned on the training set, and an anomaly score will be assigned to the sequence.

2.1 SAX : Symbolic Aggregate Approximation

This method is used to transform a continuous signal into a symbolic sequence. First, the timeserie must be Z-normalized. We divide then the data into N continuous segments, and then compute the mean value of the timeserie on each segment. It then discretizes the timserie into a sequence of symbols $\{a_1, a_2, \ldots, a_{\alpha-1}, a_\alpha\}$

The breakpoints are computed such that $\beta_1 < \beta_2 < \cdots < \beta_{\alpha-1}$ and

$$\mathbb{P}(Z < \beta_1) = \frac{1}{\alpha} \quad \mathbb{P}(Z < \beta_2) = \frac{2}{\alpha} \quad \ldots \quad \mathbb{P}(Z < \beta_{\alpha-1}) = \frac{\alpha - 1}{\alpha} \tag{1}$$

Where $Z \sim \mathcal{N}(0, 1)$. Then to each segment S_i with a mean value of $\bar{x}_i$, we attribute the symbol a_j such that $\beta_{j-1} < \bar{x}_i < \beta_j$

2.2 Kernel Based Techniques

The two methods presented below uses a kernel, ie a function that given two sequences, gives a score of how similar they are to each other. The kernel detailed in the article was the length of the longest common sequence :

$$nLCS(S_i, S_j) = \frac{|LCS(S_i, S_j)|}{\sqrt{|S_i||S_j|}} \tag{2}$$

However, there are different ways to process the results of this kernel.

- **Knearest Based** To compute the anomaly score of a new serie, we will compute the kernel score of that sample against all the serie of our dataset, take the K-th smallest socre and return the anomaly score as the inverse of that score.
- **Medoids Based** This method relies on computing a similarity matrix of all the sequence in our dataset at training time, and using this similarity matrix to find the medoids in our dataset, ie the sequences that are most "like" the other sequences in our dataset. Once obtained, the anomaly score is assigned by giving the kernel distance between the sample data and the medoid that is closed to it.

2.3 Window Based Techniques

These methods cuts the sequences in the dataset into smaller chunks and scores each of these chunks. The global anomalous score of the sequence is then a function (mean, max, a weighted sum) of the scores on each window.

- **Lookahead technique** [3] Computes the frequences of occurance of a certain pair in the test dataset and scores novel data on the frequency of ocurrance
- **Normal Dictionary** [4] stores the frequency of appearance of a given window in the training set, and returns the anomaly score of any given widnow as the inverse of that frequency.
- **OneClassSVM** [5] trains a OneClassSVM on the one hot encoded discretized data and returns the anomaly score of that SVM.

2.4 Markov Based Techniques

The Markov-based techniques rely on modeling the probabilistic transitions between symbols in a sequence. In our experiment, we covered the following ones :

- **Fixed Markovian techniques** uses a fixed-length history k to estimate the conditional probability of a symbol s_{q_i} in the test sequence as:

$$P(s_{q_i} \mid s_{q_{i-k}}, \ldots, s_{q_{i-1}}) = \frac{f(s_{q_{i-k}}, \ldots, s_{q_{i-1}}, s_{q_i})}{f(s_{q_{i-k}}, \ldots, s_{q_{i-1}})} \tag{3}$$

  where $f(s_{q_{i-k}}, \ldots, s_{q_{i-1}}, s_{q_i})$ is the frequency of the subsequence $[s_{q_{i-k}}, \ldots, s_{q_{i-1}}, s_{q_i}]$ observed in the training sequences, and $f(s_{q_{i-k}}, \ldots, s_{q_{i-1}})$ is the frequency of $[s_{q_{i-k}}, \ldots, s_{q_{i-1}}]$. An anomaly score is then computed using the estimated probability given the test sequence.

- **Variable Markovian techniques** addresses the limitations of the fixed approach by allowing the history size to vary. For this, we use a Probabilistic Suffix Trees (PST) [6]. Each node in the PST represents a variable-length context (subsequence) and tracks counts of symbols following it. The conditional probability is then computed by starting from the longest available context and moving to shorter ones until a match is found. We then compute the anomaly score for a test sequence S_q using a log scale.

$$L(S_q) = \frac{1}{|S_q|} \sum_{i=1}^{|S_q|} \log P(s_{q_i} \mid s_{q_{i-j+1}}, \ldots, s_{q_{i-1}}) \tag{4}$$

  where $p(s_{q_i})$ is the computed conditional probability of the symbol s_{q_i} and $j \leq k$.

- **Sparse Markovian Techniques** generalize the concept of context used in fixed and variable Markov models by conditioning on a sparse history instead of immediately preceding symbols. This technique introduces two approaches:

  1. **Sparse Markov Transducers (SMT)** [7] creates a forest of PSTs by allowing wildcards (X) in the context. The conditional probabilities are then estimated by traversing the SMT and matching the context, including wildcards, to predict the next symbol.

  2. **RIPPER-based Sparse Markov Techniques** [8] learns rules to predict the next symbol based on the context using a Decision Classifier.

3 Data

3.1 Synthetic Data Experiment for Markov

Due to the lack of anotated symbolic dataset online, we decided to generate some synthetic data to prove the performance of our Markov Based techniques. For different size of the transition matrix, we generated a train set of sequence generated by a markov process with a random transition matrix, which we call the **baseline matrix**, and a test set composed at 50 percent of sequence generated by the baseline matrix and 50 percent generated by 5 other randomly generated transition matrices, called **anomalous matrix** at 10 percent each. We also measured how different the matrices are to each other using the L1 norm and plotted the results with MDS 6, and analyzed coefficient by coefficient with 7.

3.2 Dodgers Dataset

For the second experiment, we wanted to test our methods on real data. We therefore fetched the dodgers dataset online .[1] This dataset contains a unique continuous time series annotated in the points where an anomalous point is being observed (see Appendix B). Before applying the SAX technique, we transformed our initial sequence into multiple fixed-length sequences using a sliding window, we then label each sequence depending if or not it contains at least one single anomalous point (see Appendix C). We then apply the SAX algorithm on the sequences extracted with a word size of 10 and an alphabet size of 7. The figure below shows a normal sequence and an anomalous one.

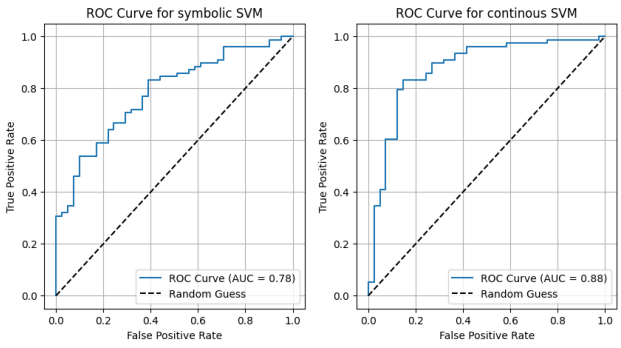
(a) Normal sequence.

(b) Anomalous sequence.

Figure 1: Visualization of the SAX algorithm applied to two sequences

3.2.1 Discretization impact on the intial dataset

Before moving forward, we thought it would be interesting to inspect the impact of the discretization on the dataset. To do so, we compared the amount of information contained in the first few eigenvalues of a PCA on the symbolic data versus the original continuous data (see Appendix D). SAX decorrelated the original continous dataset. Additionally, we tried to quantify the separability of our classes before and after applying the SAX algorithm by training a SVM on the dataset and see its AUC score. We can clearly see the decrease in the AUC after discretization, which indicates that the anomaly detection is harder : we lost some information, which is coherent with the averaging in the SAX method.



Inspection using SVM

We can see from the results that by going from continuous to discrete results in a significant loss in the information contained in the signal.

[1] Be careful it downloads the file automatically : link.

4 Results

4.1 Kernel and Window-Based Results

We use the methods proposed by the article to predict the sequences that are anomalous in our dataset. We have then compared the performances of every technique using the AUC score. The results are presented in the table in Appendix E

We can see that **kernel-based techniques** outperform window-based methods. In particular, the **K-Nearest Neighbors Kernel** achieves the highest AUC score (0.74). On the other hand, the **Medoids Kernel** performs slightly worse, probably since it relies on representative medoids, which may not be enough for our dataset.

Window-based techniques like the **Lookahead** and **Normal Dictionary** perform poorly, likely because they fail to capture the underlying structure of our dataset. Overall, the success of the kernel methods shows how powerful the distance-based methods are when combined with kernels to capture the similarities between sequences of complex representations.

4.2 Influence of the SAX parameters on the results

We decided to see the impact the parameters of the SAX algorithm have on the differents techniques. More precisely, we choosed different values of the alphabet size and the word size and ran the *K-nearest kernel method* since it was the one with the highest performance. The results are shown in the heatmap in Appendix F.

The results reveal that **higher alphabet sizes lead to better AUC performance**. This can be explained by SAX's ability to represent more nuanced transitions within the signal when the alphabet size increases. A larger alphabet size allows us to discretize the data into finer symbolic segments, meaning a higher chance of detecting variations and anomalies in the signal.

However, this needs to be balanced with **word size**, which determines the length of symbolic subsequences. In fact, smaller word sizes (e.g., 5) lose context due to extremely short symbolic representations, and very large word sizes (e.g., 50) may over-smooth the transitions. That is why the best performing scenario is with word size 20. Therefore, the combination of higher alphabet size and moderate word size allows the SAX algorithm to balance between the granularity and context.

4.3 Markovian methods on simulated data

The result of the experiment indicated **a good ability of the Fixed Markov Based and variabe Markov Based to separate baseline from anomalous data**, with better score as the number of symbols in the sequence increased. Both models showed similar performance. A very interesting phenomena to study was also the influence of the L1 distance between the anomaly matrix and the baseline matrix on the accuracy of the separation. As this distance increase, so did the AUC score, which intuitively corresponds to the fact that our markov processes are "more" different. And of course, the AUC score increased with the number of symbols in our sequence as did the baseline anomly distance. The results of the benchmark can be found at Appendix A, while the detailed distribution and AUC score are to be found at Appendix 4 and 5

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for discrete sequences: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 823–839, 2012.

[2] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing sax: a novel symbolic representation of time series," *Data Mining and Knowledge Discovery*, vol. 15, no. 2, pp. 107–144, 2007.

[3] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff, "A sense of self for unix processes," in *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pp. 120–128, IEEE, 1996.

[4] S. A. Hofmeyr, S. Forrest, and A. Somayaji, "Intrusion detection using sequences of system calls," *Journal of Computer Security*, vol. 6, no. 3, pp. 151–180, 1998.

[5] B. Gao, H.-Y. Ma, and Y.-H. Yang, "HMMs (hidden markov models) based on anomaly intrusion detection method," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, pp. 381–385, IEEE, 2002.

[6] P. Sun, S. Chawla, and B. Arunasalam, "Mining for outliers in sequential databases," in *Proceedings of the Sixth SIAM International Conference on Data Mining*, pp. 94–105, SIAM, 2006.

[7] E. Eskin, W. Lee, and S. J. Stolfo, "Modeling system calls for intrusion detection with dynamic window sizes," in *Proceedings of the DARPA Information Survivability Conference and Exposition II (DISCEX)*, vol. 1, pp. 165–175, IEEE, 2001.

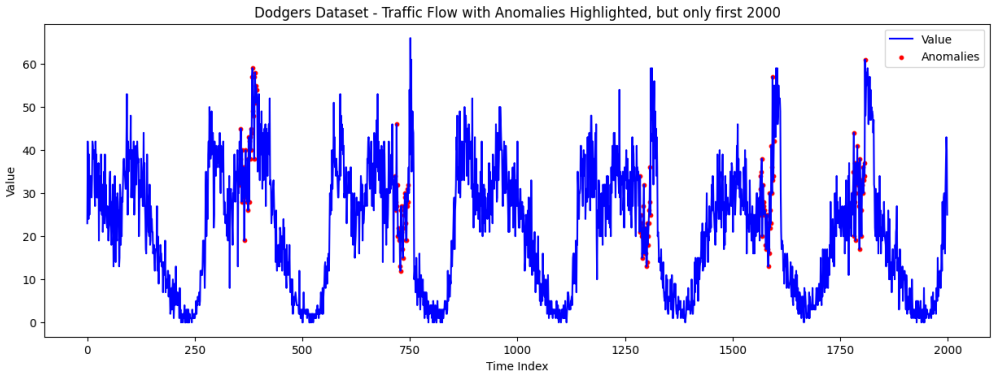
[8] W. Lee, S. J. Stolfo, and P. K. Chan, "Learning patterns from unix process execution traces for intrusion detection," in *Proceedings of the AAAI Workshop on AI Approaches to Fraud Detection and Risk Management*, pp. 50–56, AAAI, 1997.

A Markov Techniques Benchmark

Method used	Size of transition matrix	AUC SCORE	Baseline-Anomaly Matrix distance
Fixed Based	3	0.848	0.696
Fixed Based	5	0.943	0.682
Variable Based	3	0.834	0.696
Variable Based	5	1.00	0.682

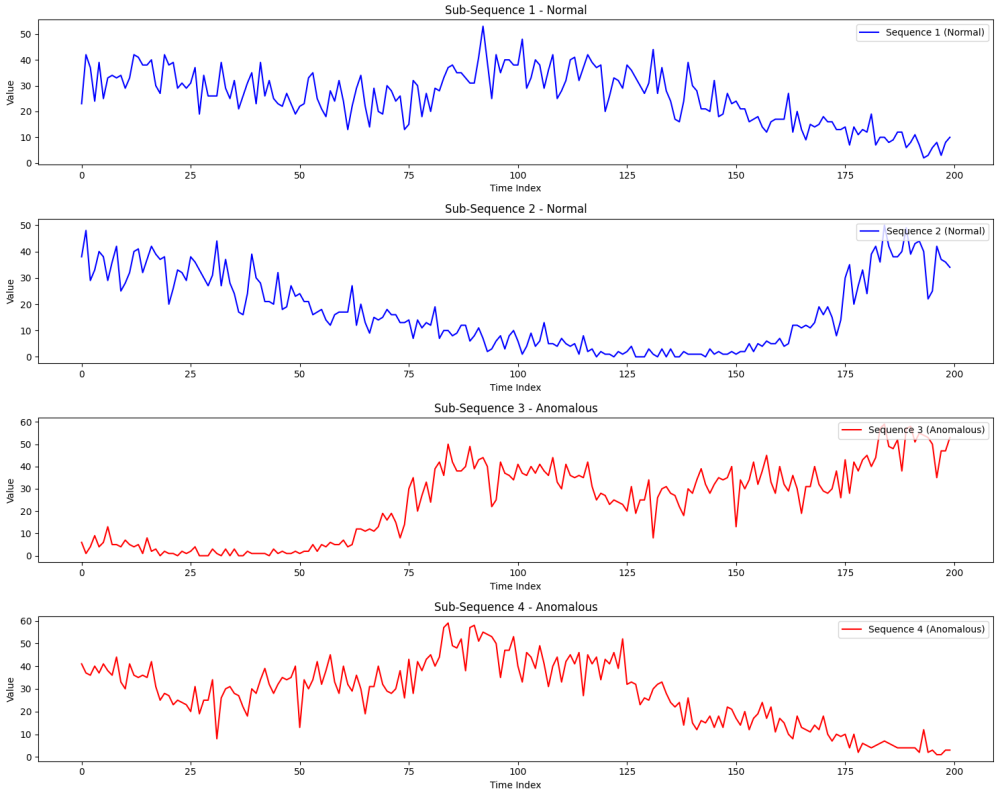
Table 1: Markov Based Techniques Benchmark Results
On the generated dataset, the two methdos performed exactly the same, however on different dataset slight variation in the AUC score can appear. As you will see in the notebook. The Baseline-Anomaly distances indicates the l1 distance between the baseline transition matrix and the anomalous transition matrix.

B Initial Dataset



Visualization of the initial dataset

C Extracted sequences



Visualization 4 examples of sequences

D Inspection of the impact of the SAX algorithm

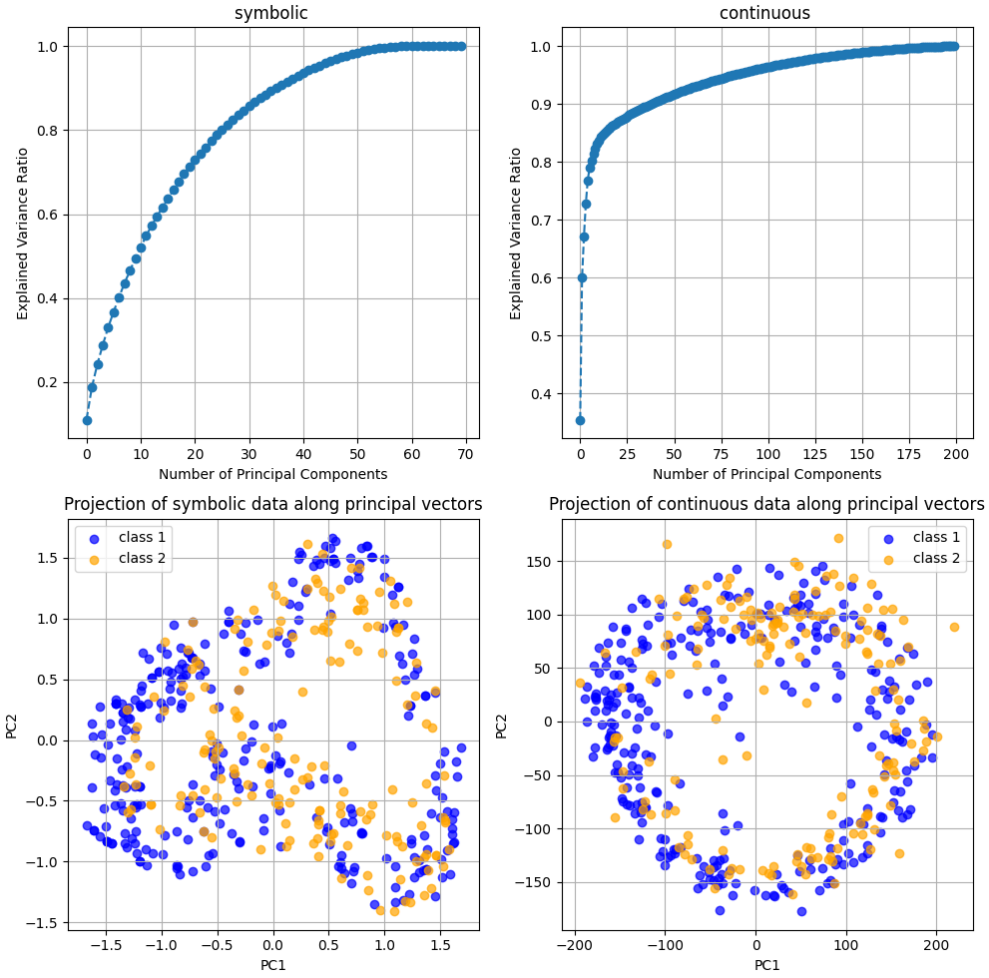


Figure 2: Impact of the SAX algorithm on the data structure
We notice that the SAX techniques is significantly decorrelating our data from the orginal continous timeserie. The projections along the first two components of our PCA do not seem to indicate interesting separability at low dimension.

E Kernel and Window based results

Method	KNN Kernel	Medoids Kernel	Lookahead	Normal Dict	Unsupervised SVM
AUC	0.74	0.67	0.40	0.45	0.42

Table 2: AUC Scores for Kernel-Based and Window-Based Techniques

F SAX parameters influence

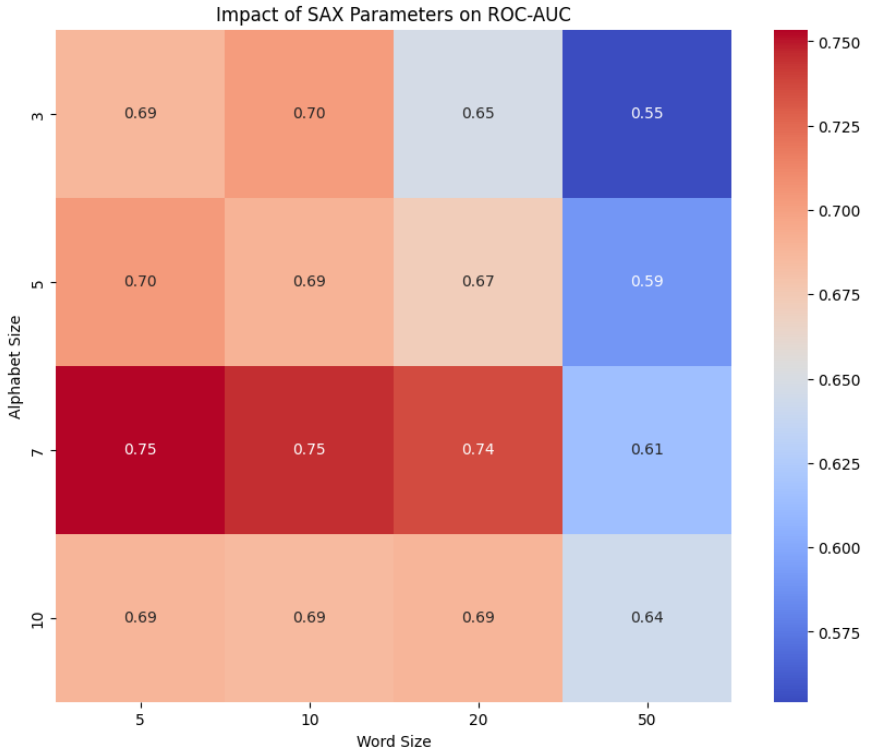


Figure 3: SAX parameter influence

G Fixed Markov Techniques : Benchmark Figure

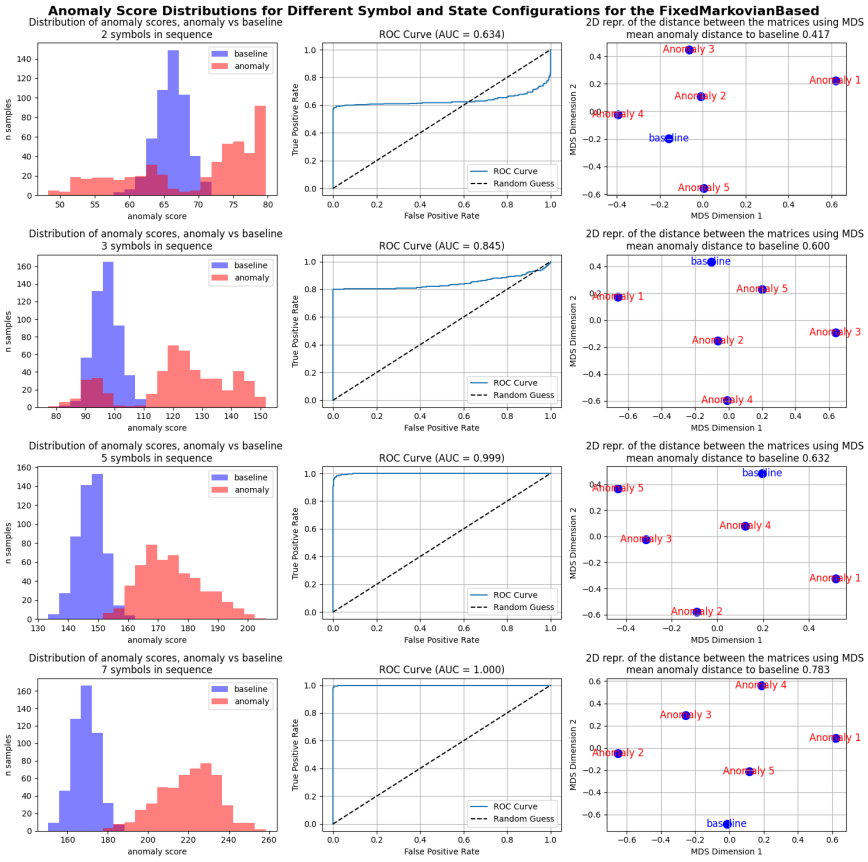


Figure 4: Anomaly Score distribution for different symbols of the Fixed Markov Based model The leftmost figure is the distribution of the anomaly score for the baseline sequences and the anomalous sequences in the test set. The rightmost figure is a 2D representation of the distance between the transition matrices that were used to generate the dataset. The technique used to obtain this graph from a distance matrix is called MDS.

H Variable Markov Techniques : Benchmark Figure

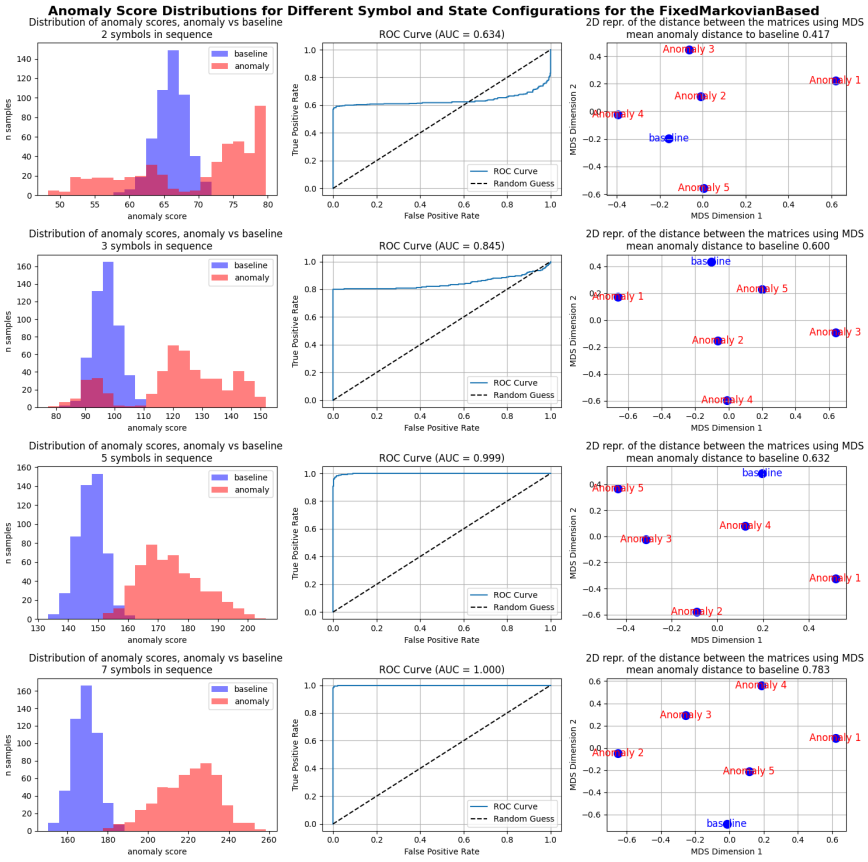


Figure 5: Anomaly Score distribution for different symbols of the Variable Markov Based model

I MDS Visualization, closer look

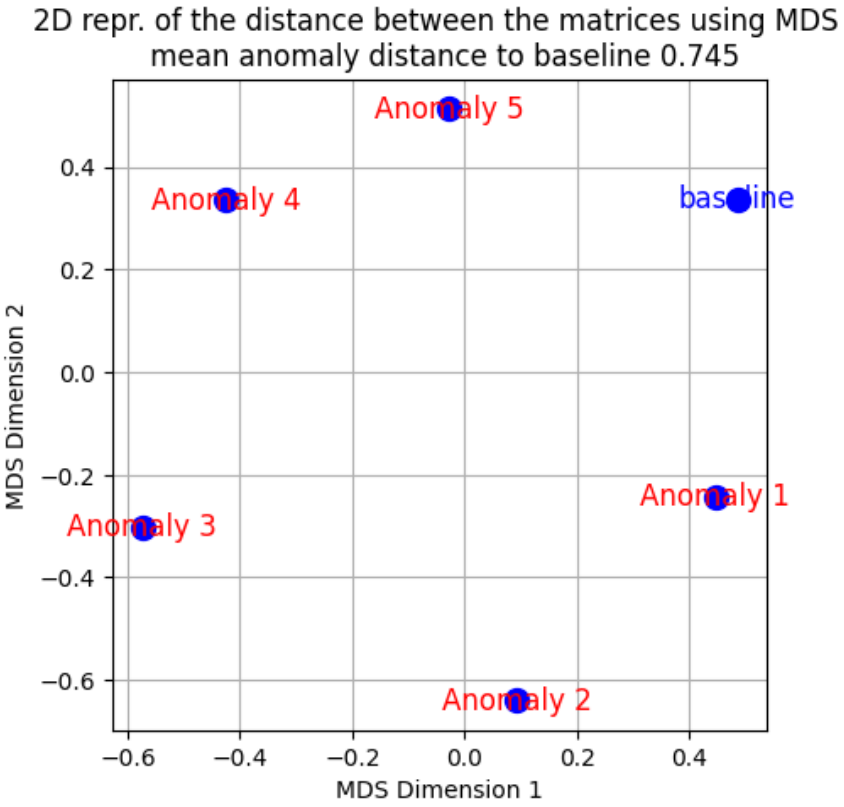


Figure 6: MDS visualisation of the distance between the transition matrix of a synthetic dataset. The MDS visualisation is computed from a distance matrix of the six transition matrices which we used to generate the markov chains in the test set.

J Comparison of anomalous transition matrices to baseline, in generating synthetic data

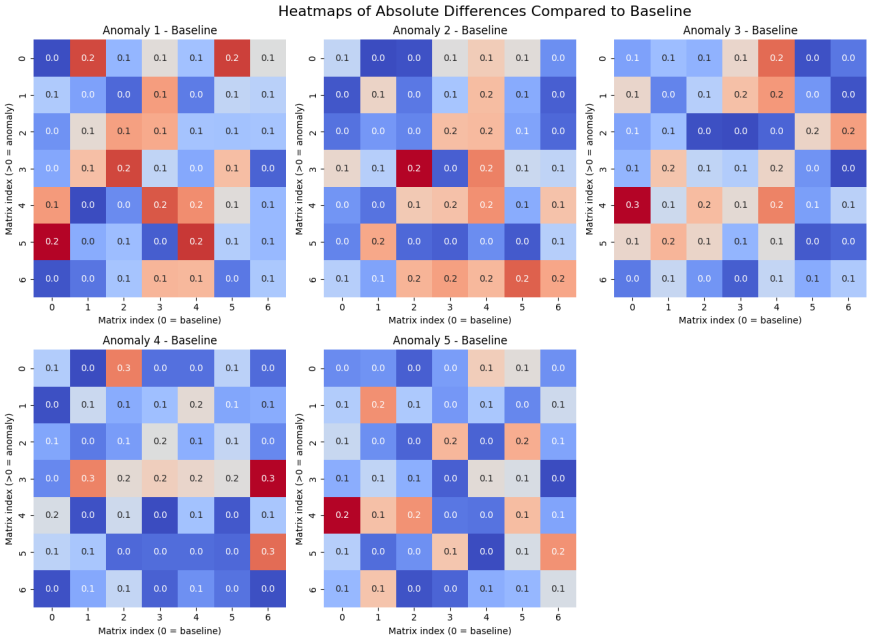


Figure 7: heatmap of the l1 difference in coefficient of the anomalous transition matrix compared to the baseline

We notice that the random generation tends to result in one coefficient being significantly different than the other in the transition matrix.