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

Melvin Gode gde.melvin@gmail.com
Samuel Sarfati samuelsarfati@outlook.com

January 8, 2025

Abstract

We based this project on Le et al. 2019 [4], a paper about a lightweight interpretable Time Series classification method involving multi-domain symbolization. We implemented every algorithm discussed in the paper from scratch and were able to test and visualize results on the classic GunPoint benchmark dataset.

1 Introduction and contributions

State of the art methods in Time Series classification include hard to interpret algorithms such as the notoriously black-box Deep Learning methods[1], computationally heavy algorithms or both at the same time.
The paper we studied for this project tries to tackle both these issues by proposing interpretable and fast running methods based on symbolic representation, symbolic feature extraction and logistic regression. It is particularly interesting in the fact that it leverages different representations of the same time series which improves on results.

Our code is available and will be joined together with this report. We re-used about 1% of pre-existing code, taking inspiration from how plots were drawn in the original paper's implementation. The rest of the code is entirely ours (with small LLM contributions for basic functions) and implements every algorithm discussed in [4].
While we tried our method on different datasets, the focus we took on the implementation part means we do not present any new case study and only one new experiment in this report. We give a basic interpretation of results on a well known dataset, demonstrating the interpretability of the method.

Contributions : Resource exploration : SS, MG; Code implementation : MG; Report redaction : MG; Search for available Data : SS, MG; Result visualization : MG; Result analysis : MG.

[1] Attention based methods can be easier to interpret but also require computation heavy training.

2 Method

The method leverages symbolic representation of time series to later apply feature selection to sub-sequences and possibly add a last logistic regression step.

2.1 Discretization

The first step of the algorithm is to symbolize our time series. The authors propose different ways of doing this : using either SAX ([5]), SFA ([6]), or a combination of both.

2.1.1 SAX

Symbolic Aggregate approXimation (SAX) proceeds by computing the Piecewise Agregate Approximation (PAA, [2]), a simple moving average of the time series to reduce its dimensionality into w values and then binning each of these values aiming for a uniform distribution across α bins (we do this simply by z-normalizing the time series and then using $\mathcal{N}(0,1)$ quantiles as breakpoints). The time series is then discretized by matching each element of its PAA to a bin to which we associate a symbol.
We also tried using empirical quantiles of the time series as breakpoints, which led to interestingly differing results.

2.1.2 SFA

Symbolic Fourier Approximation (SFA), consists first in computing the Fourier Transform of training data and keeping only the w first coefficients (separating real and imaginary in different coefficients). We then use the empirical quantiles of the distribution of *each* coefficient to create corresponding bins. After this "training" phase, we take the first w coefficient of our time series of interest, match them to their bin (α different bins) and corresponding symbols.

Finally, both methods are used with sliding windows of size l, giving a "word" representation for each window which together form a "sentence" for each entire time-series.
Both of these discretization algorithms can be used with varying parameters w, α and l, even combining the two techniques together. Creating a number of different representation of the same time series and possibly even leveraging different domains (time vs frequency). In the next step we will see how this collection of different representation is used to only select a handful of most relevant features.
We did implement both SAX and SFA, however, for interpretability reasons discussed later, we ended up only using SAX for our time series classification.

2.2 Feature Selection On Subsequences

Now that we have symbolized our time-series training set in a number of different ways, we need to learn what subsequences of symbols are most relevant in predicting class labels. To do this, the method used is SEQL ([1], [3]), which functions as follows.

SEQL is used on each representation (discretization method and parameters) independently which are later aggregated via ensemble or via logistic regression on selected features. SEQL iteratively builds features X, initially considering all singleton symbols. We also initialize a vector $\beta = 0$ which will be tyrained for outputing predictions $\hat{y} = X\beta$. We then compute the gradient of

β with respect to a classification loss function (here regularized Binomial log likelihood)[2] and only update the coordinate of β which has the largest gradient absolute value. Each considered subsequence is then expanded with the symbols following it in the training set, adding new features to X.
Now, the key phase of the algorithm resides in the fact that we can compute a *gradient upper-bound* for these new subsequences based on their prefix. With this gradient upper-bound, we can *prune* unpromising subsequences if their upper-bound is lower than the maximum gradient absolute value found at that iteration. This branch and bound pruning allows to reduce the amount of computation by a significant amount and makes the algorithm run a lot faster than if the whole sub-sequence space was explored. The algorithm then iterates on these gradient computation, feature set expansion and pruning steps until the absolute values of gradient become too small, signifying that convergence was reached.

Once the model is trained, we can output predictions by simply using $\hat{y} = X\beta$, or use logistic regression on the subsequences corresponding to features selected in the training of SEQL.
Note that this method description is for a single representation configuration. To apply this to multiple representations, logistic regression can work the same with features trained across different domains. As for the predictions using SEQL's β vectors directly, the idea is to simply ensemble these models by adding their class probability outputs together and returning the sign of this sum.

2.3 Logistic Regression

This last step is pretty self-explanatory. Once SEQL was used to find a set of useful sub-sequences, logistic regression is ran on this set to determine the best classification weights on these features.

2.4 Interpretability

The key points leading to interpretability of this method are the following two :

1. SAX is invertible : a sequence of SAX symbols corresponds to a sequence of intervals on the time-series' PAA. Therefore we can check if a subsequence on the original time-series corresponds to an important feature of the algorithm.
2. Predictions are made by linear regression : whether we use SEQL's β vector or train another one by logistic regression, the obtained coefficient directly reflect the importance of a feature in classification and thus allow us to interpret the influence of subsequences over our task.

Note that point 1. does not apply to SFA, which means we have to restrict ourselves to SAX discretization if we want to interpret the model. This leads to an important tradeoff as the authors have found better results using the two representations together : we can decide case by case if reaching maximal accuracy is most important, or if strong interpretability is a pre-requisite for a task.

[2]This is the loss function suggested in the original paper, which we also used in our implementation, note that we would have to use a different one if we wanted to predict more than two different classes.

3 Data

3.1 GunPoint Dataset

The GunPoint dataset is a benchmark time series classification dataset widely used to test algorithms due to its simplicity and clear interpretability. It consists of 50 training samples and 150 test samples, each of length 150. The dataset is binary, with two classes to predict:

- **Gun Drawn (Class 1):** Time series captured when a subject draws a gun from a holster, points it at a target and then puts it back in the holster.
- **Hand Waved (Class -1):** Time series recorded a subject has their hands to their sides, points their finger without drawing a gun, and then move their arm back down.

Each time series represents the motion of the subject's hand, encoded as a single-dimensional signal. While being a simple benchmark, the dataset still offers interesting interpretation from our models. We will therefore discuss the meaning of the data in more detail in the Result section, along with model weight visualization.

4 Results

While we limited ourselves to SAX as a discretization technique, we used three different methods on the data. First, SEQL trained on a single set of SAX parametrized representations, we trained four of these models on parameters $\alpha = 4, w = 16$ and $l = \{0.2, 0.4, 0.6, 0.8\} * L$ respectively. This is the most basic method we can have and is also resulted in varied scores, with accuracy between 0.62 and 0.87 on the test set.
Second, as suggested in the paper, we ensembled these different models by adding their "probabilities" output together and rounding to the nearest class ({-1,1}). This is our best scoring method with 0.88 test accuracy, with the ensemble outperforming each of the models it is composed of.
Finally, we also implemented *SEQL as feature selection* and ran logistic regression on the features with non-zero weights in the ensemble model. This third model scored a poor 0.63 test accuracy, almost putting at last place of all the tried models. However, severe oferfitting was observe with the model's training accuracy quickly rising to 100%. We thus tried adding L2 regularization on the weights but could not exceed 0.77 test accuracy. Ensemble SEQL remains our best model.

As mentioned in 2.1.1, we also tried the same methods with a slight tweak : using empirical quantiles as breakpoints instead of the theoretical $\mathcal{N}(0, 1)$ ones. This had unexpected influence over the results. Notably, most individual SEQL models had lower test scores, and while Ensemble SEQL's accuracy dropped to 0.84, this still resulted in a bigger performance gap between individual models and the ensemble one. But maybe the most interesting fact is that logistic regression had much better performances, scoring 0.81 without regularization and 0.83 with the best regularization parameter we found. Interpretability plots for Ensemble SEQL trained with empirical breakpoints SAX are available on figure 2.
This empirical/theoretical quantiles discussion thus remains an interesting question with empirical quantiles possibly offering better flexibility to match shorter time series with too few samples or time series with irregular value distribution which could offer worse approximation by the CLT.

Note that an upside of this method is that all models were extremely fast to train (even with somewhat ugly and un-optimized code), on top of the fact that we can visualize their weights directly on the time series.

Let us now visualize results of our best model, ensemble SEQL.

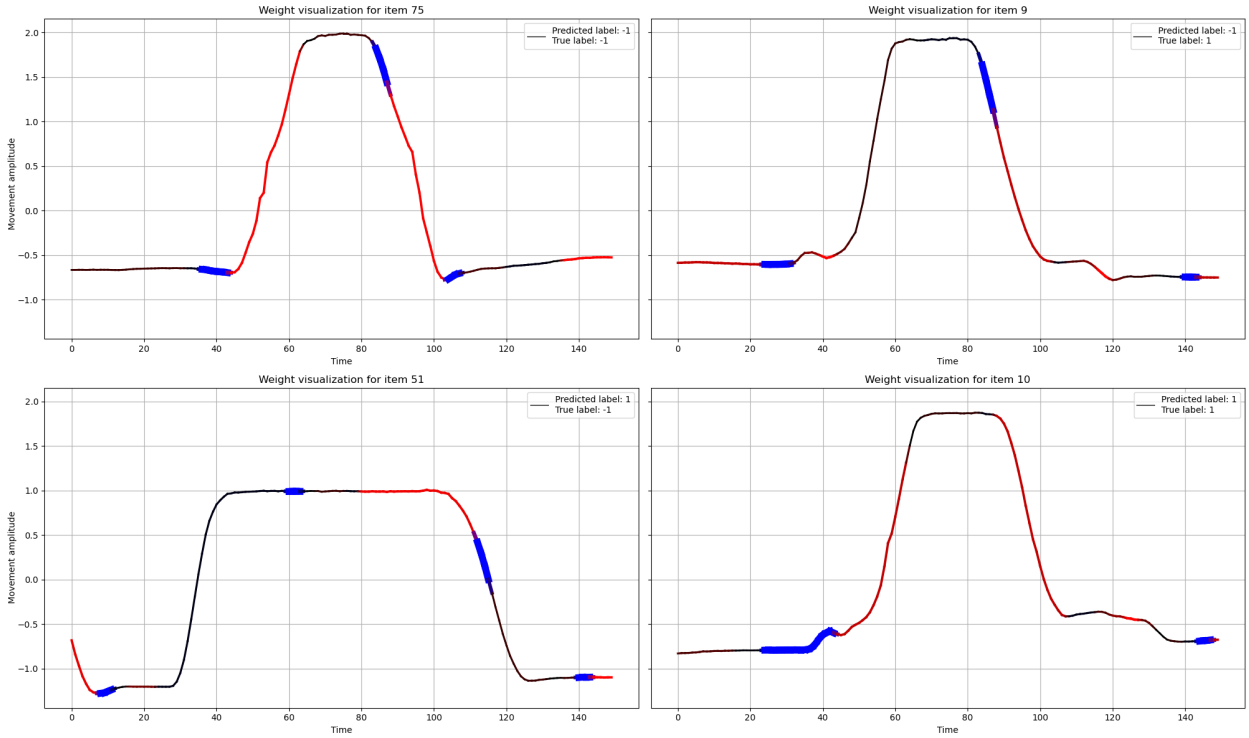

Figure 1: **Ensemble model weight visualization on GunPoint Dataset** (test set).
This plot is formed as a "confusion matrix" with true negatives and true positives along the diagonal and miss-classifications along the anti-diagonal. Red sections indicate segments contributing to classifying the time series as class 1 (gun pointing) and blue segments contribute to classifying the time series as class -1 (finger pointing).

We can observe on figure 1 what sub-sequences lead to (miss)classification of the data. First off, we can notice that the algorithm seems to focus on regions around small bumps, before and after the main peak, as well as on big rise and falls in amplitude. This is to be expected, however, it is interesting to note that little attention is usually given to the top "plateau" part where the subjects are pointing without ample movement. Indeed, we could have expected that the weight of a gun could lead to more small perturbations in the position of the hand, compared to simply pointing a finger. This may be due to the number of discretization breakpoints of SAX, which we kept to $\alpha = 4$, probably not allowing for detection of more subtle variation.
After looking at many interpretation plots on both the training and test sets, the trends that seemed to come back were that flat resting periods seemed to push for hand-pointing classification (as opposed to small bumps, pushing for gun-pointing), as the subject does not have to move its hand to put the gun in the holster. The second noted trend was that slow rises and quick falls in amplitude were associated with the gun class (resp. fast rise and slow fall for finger class), which we can attribute to the weight of the gun impacting the speed of long hand movements.

References

[1] Georgiana Ifrim and Carsten Wiuf. Bounded coordinate-descent for biological sequence classification in high dimensional predictor space. In *Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 708–716, 2011.

[2] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3:263–286, 2001.

[3] Thach Le Nguyen, Severin Gsponer, and Georgiana Ifrim. Time series classification by sequence learning in all-subsequence space. In *2017 IEEE 33rd international conference on data engineering (ICDE)*, pages 947–958. IEEE, 2017.

[4] Thach Le Nguyen, Severin Gsponer, Iulia Ilie, Martin O'reilly, and Georgiana Ifrim. Interpretable time series classification using linear models and multi-resolution multi-domain symbolic representations. *Data mining and knowledge discovery*, 33:1183–1222, 2019.

[5] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11, 2003.

[6] Patrick Schäfer and Mikael Högqvist. Sfa: a symbolic fourier approximation and index for similarity search in high dimensional datasets. In *Proceedings of the 15th international conference on extending database technology*, pages 516–527, 2012.

A Additional figure

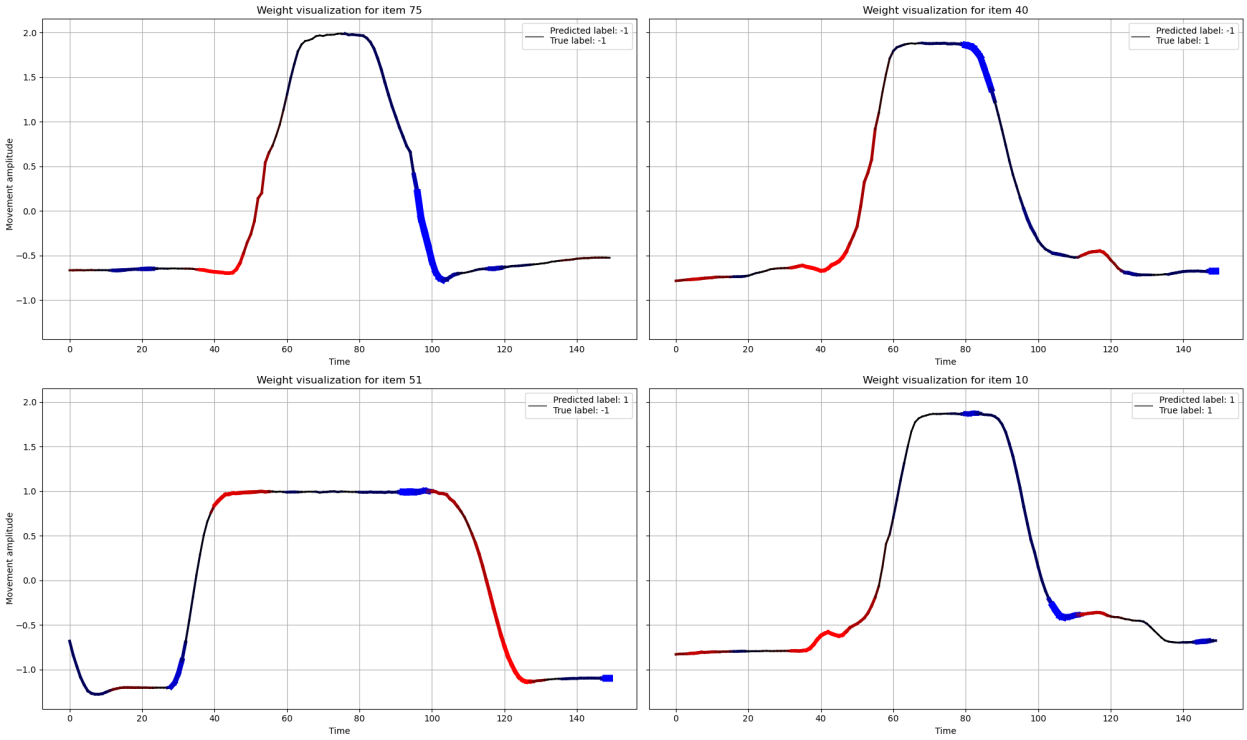

Figure 2: **Ensemble model weight visualization on GunPoint Dataset**. Model trained on empirical breakpoints SAX representations.

Plots of accuracy and max absolute gradient per training iteration are available in the notebook.