Mini-Project (ML for Time Series) - MVA 2023/2024

Baptiste CALLARD baptiste.callard@ens-paris-saclay.fr
Adhemar DE SENNEVILLE adhemar.senneville@ens-paris-saclay.fr

January 9, 2024

1 Introduction and contributions

The paper "Soft-DTW: a Differentiable Loss Function for Time-Series" [1] presents a groundbreaking approach in time-series analysis by introducing the *Soft-DTW*, a differentiable version of the Dynamic Time Warping (DTW) loss function [3], [4]. This innovation enables the use of gradient-based optimization techniques, previously inapplicable due to DTW's non-differentiable nature, thereby expanding the utility of DTW in machine learning. Soft-DTW's capability to handle time series of varying sizes and its robustness to time shifts make it highly adaptable for real-world data. It demonstrates superior performance in clustering and averaging time series, maintaining computational efficiency close to the traditional DTW. This method could significantly enhance capacities of optimization, machine learning and deep learning models dealing with time series data, offering a flexible and powerful tool for a broad spectrum of applications. Our aim is to provide a comprehensive analysis of the solutions proposed in the paper [1] from a theoretical and practical point of view, in particular by reproducing their models from scratch and conducting our own experiments. By examining the strengths and limitations of *Soft-DTW* in the context of Machine Learning, anomaly detection and from the point of view of computation time, our work contributes to the current discourse on open-source because we have proposed an optimised losses compatible with PyTorch. We will also seek to discuss the choices made in the paper and the limitations of their approach with our experiments. Code : GitHub. Our results can be launch easily : (section 4.2, 4.3 Colab), (section 4.1, 4.4 Kaggle).

In collaboration, we together comprehensively tackled various aspects of our project, encompassing re-implementation, experimentation, and report composition. Our primary task involved recreating the entire framework from scratch and do not use any other code, guided by the pseudo-code presented in "Soft-DTW: a Differentiable Loss Function for Time-Series" [1]. Our approach utilized a minimal PyTorch-based implementation. First, we reproduce the paper experiments with some variations : K-Means and barycenter. We extend the averaging of signal using *Soft-DTW* properties. Striving for innovation, we focused on anomaly detection, aware of the similarities with existing work as in [2] where they use a GAN but it is not mentioned in this work [1]. Additionally, we contributed an original analysis of the time complexity in our back-propagation implementation, comparing it with PyTorch's native functionality. With our experiments we were able to exhibit some artefact of the *Soft-DTW* and also some limitation.

2 Method

2.1 Problem definition

We denote by $\mathbf{x} = (x_1, ..., x_n) \in \mathbb{R}^{p \times n}$ and $\mathbf{y} = (y_1, ..., y_m) \in \mathbb{R}^{p \times m}$ two multi-dimensional time series. Let $\mathcal{A}_{n,m} \subset \{0,1\}^{n \times m}$ be the set of "soft path" alignment matrices that connect the coefficient (1, 1) to (n, m) by moving only with a combination of down and right directions at each step. We denote $D(\mathbf{x}, \mathbf{y}) := \left[dist(x_i, y_j)\right]_{i,j}$ the distance matrix. Given $\mathbf{x}$ and $\mathbf{y}$, solving the minimization problem of *DTW* is not differentiable due to the *Hard-min* operator. Therefore, we introduce the *Soft-min* operator defined as;

$$min^{\gamma}\{a_1, \ldots, a_n\} = -\gamma \log \left(\sum_{i=1}^{n} e^{-a_i/\gamma} \right), \quad \text{if } \gamma > 0 \quad \text{with} \quad \log \sum_i e^{z_i} = \max_j z_j + \log \sum_i e^{z_i - \max_j z_j}$$

Finally, the *Soft-DTW* is obtained by solving the problem:

$$\mathrm{DTW}_{\gamma}(\mathbf{x}, \mathbf{y}) = \min_{\pi \in \mathcal{A}_{n,m}(\mathbf{x},\mathbf{y})}{}^{\gamma} \sum_{(i,j) \in \pi} d(x_i, y_j)^2 \xrightarrow[\gamma \to 0^+]{} \mathrm{DTW}(\mathbf{x}, \mathbf{y})$$

It can be noted that as γ approaches 0^+, the smallest term among the a_i will dominate the other terms and the *soft-min* will tend towards the *hard-min*.

2.2 Differentiation for back propogation

The core of the method is the differentiability, a step that we have implemented and that we want to develop further. The *soft-DTW* is computed using a modified version of *Bellman recursion* to obtain the intermediate alignment costs matrix $\mathcal{R} := [r_{i,j}]_{i,j}$. When $\gamma > 0$, the DTW_{γ} becomes differentiable and its gradient can be easly obtained.

$$\nabla_{\mathbf{x}} DTW_{\gamma}(\mathbf{x}, \mathbf{y}) = \left(\frac{\partial D(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}} \right)^T E \quad \text{, where} \quad E = \left[\frac{\partial r_{n,m}}{\partial r_{i,j}} \right]_{i,j} \text{ and } \quad E_{n,m} = 1. \tag{1}$$

According to *Bellman recursion*, by construction, the gradient of $DTW_{\gamma}(\mathbf{x}, \mathbf{y})$ with respect to $r_{i,j}$ only depends on $r_{i+1,j}, r_{i,j+1}, r_{i+1,j+1}$.

By applying the chain rule, the quantities $\frac{\partial r_{i+1,j}}{\partial r_{i,j}}, \frac{\partial r_{i,j+1}}{\partial r_{i,j}}, \frac{\partial r_{i+1,j+1}}{\partial r_{i,j}}$ must be calculated:

$$\frac{\partial r_{n,m}}{\partial r_{i,j}} = \frac{\partial r_{n,m}}{\partial r_{i+1,j}} \frac{\partial r_{i+1,j}}{\partial r_{i,j}} + \frac{\partial r_{n,m}}{\partial r_{i,j+1}} \frac{\partial r_{i,j+1}}{\partial r_{i,j}} + \frac{\partial r_{n,m}}{\partial r_{i+1,j+1}} \frac{\partial r_{i+1,j+1}}{\partial r_{i,j}}$$

Let focus on $r_{i+1,j} = \delta_{i+1,j} + min^{\gamma}\{r_{i,j-1}, r_{i,j}, r_{i+1,j-1}\}$, we know that $\frac{\partial r_{i+1,j}}{\partial r_{i,j}} = \frac{e^{-\frac{r_{i,j}}{\gamma}}}{e^{-\frac{r_{i,j-1}}{\gamma}} + e^{-\frac{r_{i,j}}{\gamma}} + e^{-\frac{r_{i+1,j-1}}{\gamma}}}$ and thus $\gamma \log \left(\frac{\partial r_{i+1,j}}{\partial r_{i,j}} \right) = min^{\gamma}\{r_{i,j-1}, r_{i,j}, r_{i+1,j-1}\} - r_{i,j} = r_{i+1,j} - \delta_{i+1,j} - r_{i,j}$.

Therefore, we can have a way to compute recursively the back-propagation :

$$\frac{\partial r_{i+1,j}}{\partial r_{i,j}} = exp^{\frac{1}{\gamma} r_{i+1,j} - \delta_{i+1,j} - r_{i,j}} \qquad \frac{\partial r_{i,j+1}}{\partial r_{i,j}} = exp^{\frac{1}{\gamma} r_{i,j+1} - \delta_{i,j+1} - r_{i,j}} \qquad \frac{\partial r_{i+1,j+1}}{\partial r_{i,j}} = exp^{\frac{1}{\gamma} r_{i+1,j+1} - \delta_{i+1,j+1} - r_{i,j}}$$

With equation (1), we have access to $\nabla_{\mathbf{x}} DTW_{\gamma}(\mathbf{x}, \mathbf{y})$. We use these results and their pseudo code then to compute the backward.

3 Data

3.1 Objective and Selection Criteria

The primary objective in selecting the dataset was to ensure its suitability for our experiments focused on Soft Dynamic Time Warping (*Soft-DTW*). The dataset needed to fulfill several criteria:

1. Based on computational efficiency considerations, we required short time series that encapsulate meaningful structure and information. While the data could be multivariate and with many samples, the length of each time series was restricted to a maximum of 100 to facilitate numerous training and experimental iterations.
2. The dataset should predominantly feature signals where the core attributes are embedded within the shape of the data. This characteristic is crucial to leverage the advantages offered by time warping techniques.
3. For robust deep learning applications, the dataset must comprise a sufficiently large number of samples to adequately populate training, validation, and testing subsets.

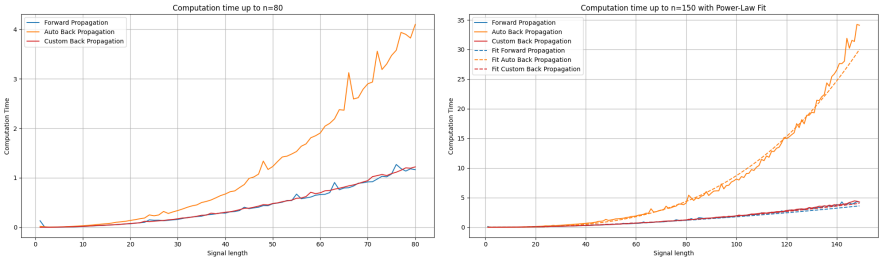
Tslearn is a Python machine learning library built for time series analysis, providing tools for time series classification, clustering, and more. We utilized *Tslearn* to identify datasets that meet our application requirements. The datasets sourced are clean, derived from real-world data, and thus did not necessitate additional preprocessing techniques. It also allows us to test our code on multiple datasets.

1. Barycenter Computation: We choose to present results on the *ECG200*, *TwoLeadECG*, *ProximalPhalanxOutlineAge-Group* dataset to exhibit interesting results.
2. KMeans Clustering: We selected *SonyAIBORobotSurface2* with short and identifiable clusters.
3. Anomaly Detection: For anomaly detection tasks, we utilized the *MiddlePhalanxOutlineCorrect* dataset, comprising 600 signals, each with a length of 80. It is well-suited for our application, featuring two distinct classes. The first class was designated as "normal" signals, and the second comprised the anomalies. Additionally, we experimented with generating a synthetic dataset using varying frequency sinusoids to enhance the interpretability of the *Soft-DTW*. We also applied our methodology to real-world datasets; however, the outcomes were less promising, likely due to the extensive length of the signals and the low quality inherent in the data.

4 Results

4.1 PyTorch implementation

We have proposed an optimised, distributed and a PyTorch fully integrated implementation of the *Soft-DTW*. We first implemented the forward so that it would be compatible with PyTorch in order to be able to use AutoGrad. However, the computation time was longer than quadratic. So we integrated our own implementation of the back-propagation into PyTorch for batches of signals $\mathbf{x}$ of the same size that could be different from targeted signals $\mathbf{y}$. Along with the parallelization on batches, we also added GPU compatibility and encapsulated *Soft-DTW* within a class structure for improved usability akin to native PyTorch loss functions such as Mean Squared Error Loss. This modernization simplifies the application of *Soft-DTW* in diverse computational contexts. In our computation time study, we study the impact of the length of the signal (n) for a fixed batch size and number of features (here 16 and 2 respectively)



(a) Up to $n = 80$ (b) Up to $n = 150$ with Power Fit laws

Figure 1: Computation Time Study : Up to $n = 80$ and Up to $n = 150$ with Power Fit laws

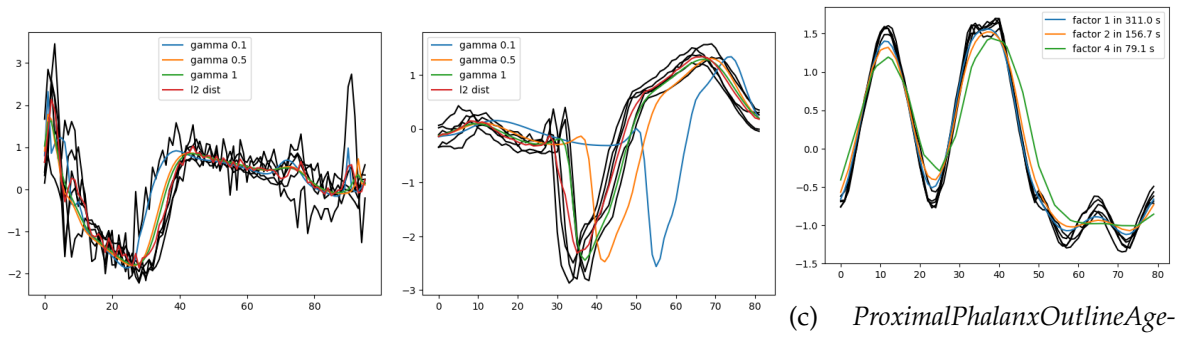
The AutoGrad takes a significantly larger time to compute. As expected back propagation and forward propagation take a similar time to compute using the custom method from the paper. However, it is still 100 times lower than MSE.

4.2 Barycenter : Averaging with the *soft-DTW*

As propose in the original paper, we want to solve the problem consiting of finding the weighted barycenter of a set of signals of non-homogeneous sizes. We define the following objective function: $\min_{\mathbf{x}\in\mathbb{R}^{p\times n}} \sum_{i=1}^{N} \frac{\lambda_i}{m_i} \mathrm{dtw}_\gamma(\mathbf{x}, \mathbf{y}_i)$ with λ the weighting and $\mathbf{m}$ the vector of signal lengths.

Overall, in figure (2) we can see that these methods effectively capture the trend. As espected, when γ increase then the barycenter become smoother and even smoother than l_2 (see. figure (2a)). In figure (2b), the *Soft-DTW* with $\gamma \in \{0.1, 0.5\}$ shows a limit (or an extension) of the method compared to l_2 . In fact, *Soft-DTW* enables to propose barycenters not aligned but close in the sense of the *Soft-DTW*.

To use MSE with signals of different size, it is possible to do the training after up/down-sampling to a fix size signals. We propose an original approach dual of this first approach using *Soft-DTW*. We decided to find a barycenter of a certain predefined size regardless of the training set size. Given a training set $\mathbf{Y}:=\{\mathbf{y}_i\}_i$ with $\mathbf{y}_i \in \mathbb{R}^{m_i\times d}$, we try to find a barycenter $\mathbf{x} \in \mathbb{R}^{n\times d}$. In our case $m_i = m$, thus we choose $n = \frac{m}{factor}$ with $factor > 0$ such that $n \in \mathbb{N}$. After convergence, we perform up-sampling using linear interpolation (see. figure (2c)). With a factor of 2, we have divided the computation time by 2. However, when we reduce the sampling too much, we lose the periodic properties (see $factor = 4$ between indexes 60-80).



(a) *ECG200* dataset (b) *TwoLeadECG* dataset (c) *ProximalPhalanxOutlineAgeGroup* dataset

Figure 2: Computation of the barycenter for a class in a different data set.

4.3 Generalization : K-Means with *soft-DTW*

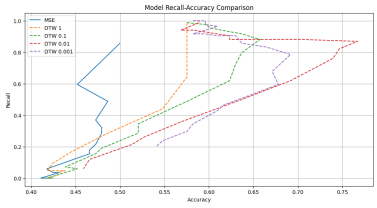
As suggested in the paper, we chose to adapt the K-Means algorithm using the differentiability of the *soft-DTW*. We denote k the number of clusters, S_i is the set of points in the i-th cluster : $\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} DTW_\gamma(\mathbf{x}, \mu_i)$ where the centroid μ_i are computed with the previous Barycenter algorithm. In the figure (4) in appendix, we set $k = 2$. We plot the barycenters of the clusters found for the *Soft-DTW* and add the barycenters with a loss l_2. We observe coherent clusters and that the barycenters are similar but smoother with *Soft-DTW*.

4.4 Anomaly detection

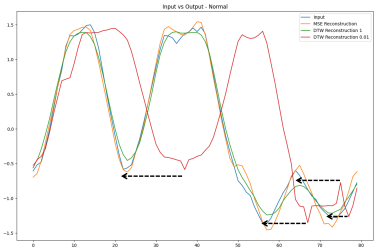
In order to benefit from the differentiability of the *Soft-DTW* as a loss function, we looked for a seq-2-seq application. We considered anomaly detection using neural networks to make use of back-propagation.

We use an Autoencoder trained exclusively on "normal" data to minimize the reconstruction error $\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = dist(\mathbf{x} - \hat{\mathbf{x}})$. Samples out of the manifold yield a reconstruction error exceeding the distribution of normal samples and can be detected with a threshold. We benchmarked the model using mean squared error (MSE) as a baseline and experimented with *Soft-DTW* values ranging from 0.001 to 1, following precedents set in literature.

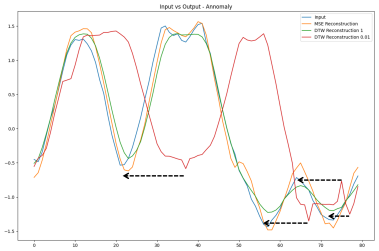
We presented our findings through plots in figure (3) the relationship between accuracy and recall across various thresholds. Notably, *Soft-DTW* demonstrated superior performance compared to the traditional mean squared error approach. Additionally, when reconstructing anomalies alongside standard signals, we observed that the model exploits the *Soft-DTW* loss by creating spikes at strategic locations within the signal (see. (3b), (3c)), effectively reducing the loss.



(a) Recall/Accuracy for varying γ



(b) Reconstruction for normal



(c) Reconstruction for anomaly

Figure 3: Experience about anomaly detection

5 Conclusion

The differentiability of *Soft-DTW* allows for its use as a loss function, enabling the application of gradient descent and thus its utilization in optimization problems in Machine and Deep Learning. These include applications such as Barycenter estimation, KMEANS, KNN, Autoencoders, RNNs, etc. However, the use of *Soft-DTW* in real scenarios can quickly be challenging due to the computational time required for the forward and backward processes. Training algorithms in machine learning necessitates a large dataset, but as the volume of data increases, the computational time dramatically explodes. Although we have implemented an optimized version of *Soft-DTW*, we worked with batches of the same size for $\mathbf{x} \in \mathbb{R}^{n \times d}$ and $\mathbf{y} \in \mathbb{R}^{m \times d}$, which limited our applications. A future avenue of work involves making this approach generic and developing a Python library including Sakoe-Chiba band. We also observed artifacts in the reconstruction of signals using Soft-DTW Loss, such as spikes and time shifts.

6 Appendix

6.1 Generalization : K-Means with soft-DTW

We display the results from section 4.3.

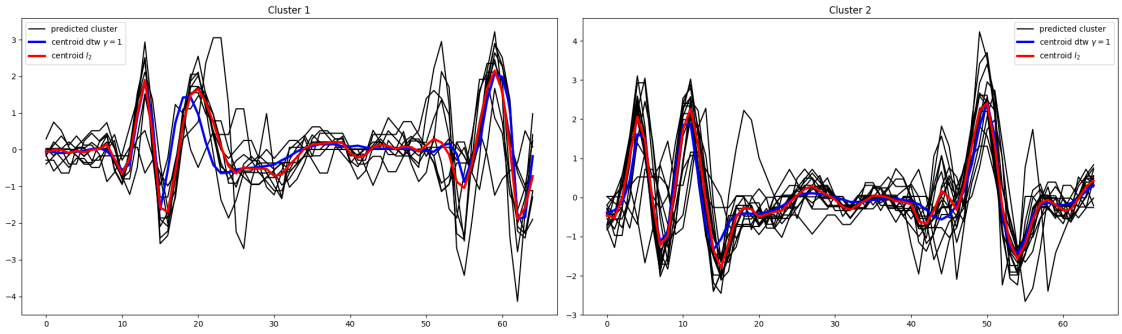


Figure 4: K-Means wtih *Soft-DTW* on dataset *SonyAIBORobotSurface2*

References

[1] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International conference on machine learning*, pages 894–903. PMLR, 2017.

[2] Shenghua Liu, Bin Zhou, Quan Ding, Bryan Hooi, Zhengbo Zhang, Huawei Shen, and Xueqi Cheng. Time series anomaly detection with adversarial reconstruction networks. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):4293–4306, 2022.

[3] Hiroaki Sakoe. Dynamic-programming approach to continuous speech recognition. In *1971 Proc. the International Congress of Acoustics, Budapest*, 1971.

[4] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.