Autonomous Driving Validation with Model-Based Dictionary Clustering

Etienne Goffinet[1,2](✉), Mustapha Lebbah[1], Hanane Azzag[1], and Loic Giraldi[2]

[1] Sorbonne Paris-Nord University, LIPN-UMR 7030
99 Avenue Jean Baptiste Clément, Villetaneuse, France
[2] Groupe Renault SAS, Avenue du Golf, Guyancourt, France

Abstract. Validation of autonomous driving systems remains one of the biggest challenges that car manufacturers must tackle in order to provide safe driverless cars. The complexity of this task stems from several factors: the multiplicity of vehicles, embedded systems, use cases, and the high level of reliability that is required for the driving system to be at least as safe as a human driver. In order to circumvent these issues, large scale simulation that reproduces physical conditions is intensively used to test driverless cars. Therefore, this validation step produces a massive amount of data that needs to be processed. In this paper, we present a new method applied to time-series produced by autonomous driving numerical simulations. It is a dictionary-based method that consists in three steps: automatic segmentation of each time-series, regime dictionary construction, and clustering of produced categorical sequences. We present the time-series specific structure and the proposed method's advantages for processing such data, compared to state-of-the-art reference methods.

Keywords: Autonomous car development · Time series clustering · Mixture models · Dictionary models

1 Introduction

Autonomous car development remains a challenge for car manufacturers. One way to solve this problem is to develop driver assistance systems that are gradually introduced in new car models. This development requires a large amount of data, of good quality, and in large quantities. To provide such data, *Groupe Renault* has made the technical choice to invest in driving simulation technology. This choice led to the development of a dedicated simulation platform that reproduces driving conditions based on car physics, driver behavior, and interaction with a parameterizable environment. This tool allows us to overcome physical simulation limits and to assess an autonomous control law with greater certainty. The simulation process outputs a large amount of information in the form of multivariate time-series. Data size, complexity, and dimensions are considerable: for the validation of the control law, the order of magnitude is $\mathcal{O}(10^6)$ simulations, with $\mathcal{O}(10^3)$ sensors, each recording at $\mathcal{O}(10^4)$ time steps.

In total, the validation of a use case requires the production of more than $\mathcal{O}(10^{13})$ data points.

Specific visualization methods are needed to analyze such data. Clustering is a first approach to tackle this problem, which consists in the automatic grouping of "similar" observations into homogeneous groups (clusters). With the help of these tools, the expert has a way to discriminate the time series but also the associated parameters. He can then isolate the effects of the control law parameters and adjust them adequately. Time-series clustering has been widely studied in the past decades. Many dedicated methods have been proposed, each based on specific assumptions on the underlying data structure. These assumptions are crucial as they determine both the clustering results and their interpretability.

In this paper, we present a new method applied to time-series produced by autonomous driving numerical simulations. It is a dictionary-based method that consists of three steps: automatic segmentation of each time-series, regime dictionary construction, and clustering of produced categorical sequences. In this paper's second section, we present the detailed simulation method and the time series structure. In the second part, we discuss the existing approaches and describe our contribution. In the third section, we present the results obtained on public datasets and on an industrial use case: the Autonomous Emergency Braking (AEB) system validation. Finally, we conclude on our method's capabilities and perspectives.

2 Simulating Autonomous Behaviour

Validating an autonomous driving rule is a complicated task, that was for a long time addressed with on-track simulations. The numerical simulation approach allows overcoming the limits of these physical simulations. A large scale simulation reproduces physical conditions is intensively used to test driverless cars. Therefore, this validation step produces a massive amount of time series that needs to be processed.

2.1 Numerical Simulation assets

Several aspects motivate the use of an autonomous behavior simulation platform. The first motivation is the physical simulation cost, which requires infrastructure, equipment management, and significant human intervention. One digital simulation is estimated 10,000 times cheaper than its physical counterpart. The savings achieved through the use of digital simulation add up to millions of euros. The second motivation comes from the fact that physical simulation is the measurement uncertainty: sensors accuracy, but also initial conditions setting.

Another major disadvantage of physical simulation is the impossibility of producing enough data. A validation objective may be the assessment of vehicle incident odds (e.g. $< 10^{-8}$ incidents per hour). With a classical sampling method, estimating such probability would require running prototypes over hundreds of millions of kilometers.

Even if such a large amount of real-life data were available, as is the case in some data science application fields, there would be no guarantees of the data quality or value. In our case, this value lies in the specific driving situation in which to test the control law reaction. These situations are rarely observable in reality, such as the ones of an emergency braking.

2.2 I/O of the Simulation Platform

Assessing a control law reliability requires taking into account every possibility, even the rarest cases. Therefore, validating such system is only feasible with accurate control of each simulation context, operated by a set of parameters divided into five categories:

– Environment parameters: road characteristics, weather conditions, but also driver behavior (cautious or sporty, cooperative or competitive).
– Car physics: weight distribution, engine capacities, etc.
– Sensors to be recorded, including the frequency of observation.
– Control law: triggers reacting to specific conditions (e.g. in the case of emergency braking, the distance to the next car) and with parameterizable effects on the vehicle (e.g. the braking intensity).
– Scenario: a sequence of phases followed by the driver and which puts the car in an experimental context (e.g. reaching a specific speed, then a cruise speed for a specific period).

Several hundreds of parameters, in total, interact to generate simulations and produce time-series. In some use cases, field experts may provide additional labels to help the classification task. However, because of the variety and complexity of the driving situations, drawing up an exhaustive list of the labels is an arduous task. The supervised approach is, therefore, unpracticable.

The scenario is the main factor in time-series construction. Other factors have secondary effects and mainly influence the duration and intensity of the phases (e.g. time to reach cruise speed, braking power, etc.). Therefore, even if several time-series originate from the same scenario, their phases may not be synchronous. Another consequence is that the output time-series differ in length.

2.3 AEB Use Case

In the majority of use cases, the autonomous driving simulation produces a large amount of unlabeled data. To validate our clustering approach, we apply it to the specific AEB use case, in which a ground truth is easily producible. In this situation, the goal is to test the reactions of a car (usually called Ego) equipped with the control law. Ego runs in a straight line towards another vehicle, which moves in the same direction but at a slower pace. We expect the target vehicle detection to trigger the control law, which in turn provokes an emergency braking. The control law objective is to prevent the collision. Three cases can arise:

- The control law is not triggered.
- Target is detected, but braking cannot avoid the collision.
- The target is detected, and braking prevents the collision.

In this illustrating use case, field experts visually assessed the different situations to provide a ground truth. The time series dataset is partitioned in 3 classes according to these labels, depicted in figure 1.

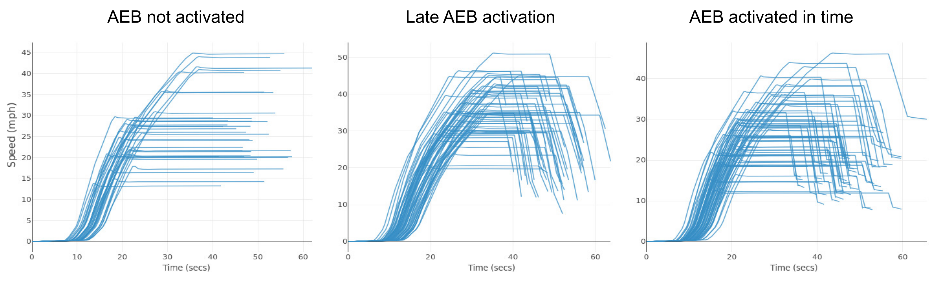

Fig. 1. Time series distribution partitioned by ground truth label.

In order to address this data structure, we developed a clustering workflow independent from the time-series length or regime synchronicity. It relies on the hypothesis of a latent scenario presence.

3 Related Work

Time series Unsupervised classification (or clustering) is a method that aims to partition a dataset into groups of "similar" temporal observations, which is the first step toward understanding its structure. Defining the similarity between times-series is a crucial point as it determines both the clustering results and their interpretations.

3.1 Distance-based clustering

The Euclidean distance is one of the most popular for this task. In this case we handle time series as n-size vectors. In practice, this metric is not the most practical as it does not take into account the temporal information and requires aligned series and of equal length.

The Dynamic Time Warping (DTW) [18] measure is another typically relevant metric in the presence of local or uniform temporal scaling (a.k.a. warping). Inspired by the edit distance (used in the context of string comparison), DTW is a measure of the effort required to match two series point-to-point. Although quite resource-intensive originally ($O(n^2)$ complexity), improved version developments over the years have allowed this approach to remain a reference in the domain [11, 19].

As presented by [1], choosing a distance is equivalent to determining the invariances to be used for cluster construction. For instance, DTW-based clustering relates to warping invariance. It is, first of all, a hypothesis on the global data structure and a way to cluster shapes. In our case, this approach would not exploit nor conserve the regimes' information. Feature-based clustering makes different assumptions.

3.2 Feature-based clustering

The feature-based approach is about designing a way to transform time series into condensed representations. The hypothesis is that this transformation keeps the informative aspects of the data. Two situations can be distinguished: the first when the transformation process is known, the second when it is estimated based on an external criterion (risk, measure, or model assumptions). Either way, this method requires prior knowledge on the time series. The dictionary-based methods family, as proposed by [15] and [21], are based on feature extraction by uniform time step segmentation and are representative of the first case. This approach was first appealing in our application as it allows, to a certain extent, the comparison of similar segments between time series. However, it requires setting arbitrary parameters (including, but not limited to, the dictionary size and the uniform segmentation time step), which is not possible in an unsupervised context. The Time-series Forest method [7] illustrates perfectly the other situation, where feature extraction relies on the supervision of a score based on Entropy and distance. Deep Learning can also be used in this context, as in [13] where the extraction is based on the reconstruction error.

In this application, we make full use of the hidden scenario hypothesis and apply a specific case of feature-based clustering method: a regime-changing time series approach.

3.3 Regime-changing Time Series Clustering

AEB use case time series are the result of the chaining of distinct phases, also known as regimes. Provided the ability to detect those regimes, it is possible to use their estimated distribution (order, frequency, amplitude. . .) to characterize the observations and discriminate them. During the last decades, several papers have been proposed to detect optimal regime change points. Those methods sum up to piece-wise polynomial regression models. The common strategy relies on optimizing an approximation error in different ways: sliding windows of increasing size as in [10] and [8], by dynamic programming as in [12], Hidden Markov models in [9] or by regression mixture models in [3]. We selected this last model for two reasons: on the one hand, the benefits of using a mixture model (confidence intervals, model selection strategy. . .) and on the other hand, the particular performances of this model compared to hidden Markov model approaches and its computational efficiency compared to dynamic programming methods [3, 4].

In *mixRHLP* from [20], the same author combines the piece-wise regression model in a finite mixture to construct a one-step model-based clustering method. The proposed approach aims at regrouping time series with common regimes cut-points. *mixRHLP* also assumes that the number of regimes is known. These assumptions are not not the case in our approach.

Our contribution is an attempt to adapt *mixRHLP* to our constraints. It consists of a three-steps workflow with the addition of an original strategy of segmentation model selection. In the first step, we apply Individual time-series segmentation with a polynomial regression mixture. In the second step we build a standard dictionary of regimes by clustering the extracted segments. The clustering of these sequences using Levenshtein distance in categorical sequence space produces the final result. Our method, called *SDLHC* for Segmentation, Dictionary construction, Levenshtein Hierarchical Clustering, has the following advantages:

– Clustering based on regime detection is intuitive and easily interpretable by experts.
– The method can be applied to a dataset of time-series with unequal lengths. Moreover, it is independent of the time-series synchronicity and the regime's moment of appearance synchronicity.
– The segmentation phase can be applied independently on each time-series, which makes the computation an embarrassingly parallel task. This step drastically reduces the data dimension.
– Our segmentation strategy optimizes automatically both the number of segments and polynomial regression on each segment, which allows to get rid of assumptions on the number of regimes and on their optimal order of polynomial regression.

4 A three-step time-series clustering algorithm (SDLHC)

The method *SDLHC* is composed of three steps: segmentation, dictionary construction, and categorical sequence clustering. The first two steps are addressed with the mixture model approach.

4.1 Segmenting time-series with a mixture of polynomial regressions

The Regression with Hidden Logistic Process from [3] is based on a polynomial regression model mixture, with time-dependent proportions following a hidden logistic process. Given a time-series $x = (x_t)_T$ and $\phi = (\phi_s(t) = t^s)_{s \in 0,\dots,S}$ a polynomial basis of size $S \in \mathbb{N}$ (e.g. monomial basis, Legendre basis, Fourier basis, etc.). A Polynomial Regression Model (PRM) of sequence x in the basis ϕ is defined by

$$\tilde{x} = \sum_{s=1}^{S} \beta_s \phi_s(t) + \sigma^2 \epsilon,$$

with $(\beta_s)_{s\in 1,\dots,S} \in \mathbb{R}^S$, $\sigma \in \mathbb{R}_*^+$ and $\epsilon \sim \mathcal{N}(0,1)$. These PRMs are the segmentation mixture model components.

Given a number of clusters $K \in \mathbb{N}$, let $z = (z_t)_{t\in T}$ be the elements $x = (x_t)_{t\in T}$ cluster membership. At a given time t, z_t follows a Multinomial distribution with parameters $\pi(t) = (\pi_k(t))_{k\in 1,\dots,K}$. The distribution of x at time t is defined by

$$p(x_t) = \sum_{k=1}^{K} \pi_k(t) f_{\theta_k}(x_t),$$

and the sequence x log-likelihood,

$$l(x;\theta) = \sum_{t=1}^{T} log\left(\sum_{k=1}^{K} \pi_k(t) f_{\theta_k}(x_t)\right), \tag{1}$$

with $f_{\theta_k}(x_t)$ the density associated to a PRM component. The varying proportions $\pi_k(t)$ can be seen as the parameters of a Multinomial distribution followed by the clusters memberships at a given time t. These proportions vary according to a logistic process. More formally, for $k \in \{1,\dots,K\}$ and $t \in T$,

$$\pi_k(t) = p(z_t = k) = \frac{\exp(\sum_{s=1}^{S} w_{k,s}\phi_s(t))}{\sum_{h=1}^{K} \exp(\sum_{s=1}^{S} w_{h,s}\phi_s(t))}, \tag{2}$$

with $w_k = (w_{k,s})_{s\in 1,\dots,S}$ the associated model parameters. In the following paragraphs, we denote by w the set of parameters $(w_k)_{k\in 1,\dots,K}$. The complete set of parameters is finally $\theta = (w, \beta, \sigma)$. The log-likelihood (1) optimization requires a specific version of the Expectation Maximization (EM) algorithm described in [6]. The EM algorithm is a standard algorithm for likelihood maximization in the presence of incomplete data. In our case, these missing data are the cluster's membership, denoted by z (the hidden variable). It is an iterative algorithm, each iteration composed of two steps.

Expectation step (E) : Given the parameters θ, the first step of the EM algorithm consists in optimizing the complete log-likelihood defined as below:

$$\begin{aligned}\mathbb{E}_{x,\theta}\left[l(x,z;\theta)\right] &= \mathbb{E}_{x,\theta}\left[\sum_{i=1}^{n}\sum_{k=1}^{K} \mathbb{I}_{z_i=k} log\left(p(x_i, z_i = k;\theta)\right)\right] \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{i,k} log\left(\pi_k f_{\theta_k}(x_i)\right). \end{aligned} \tag{3}$$

The development of the equation (3) shows that this step is simplified to the estimation of $\tau_{i,k} = p(z_i = k|x_i;\theta)$, the posterior distribution of z conditionally to x. The Bayes theorem gives the following estimation of this quantity:

$$\tau_{i,k} = p(z_i = k|x_i;\theta) = \frac{p(z_i = k, x_i;\theta)}{p(x_i)}$$
$$= \frac{\pi_k f_{\theta_k}(x_i)}{\sum_{h=1}^{K} \pi_h f_{\theta_h}(x_i)}. \tag{4}$$

Maximization step (M) : At each iteration, the model parameters are updated during the Maximization step. In this phase, the following decomposition of the complete log-likelihood expectation is maximized:

$$\begin{aligned}\mathbb{E}_{x,\theta}\left[l(x,z;\theta)\right] &= \sum_{t=1}^{T}\sum_{k=1}^{K} \tau_{t,k} log\left(\pi_k f_{\theta_k,t}(x_t)\right) \\ &= \sum_{t=1}^{T}\sum_{k=1}^{K} \tau_{t,k} log \pi_k + \sum_{t=1}^{T}\sum_{k=1}^{K} \tau_{t,k} log f_{\theta_k,t}(x_t) \\ &= Q_1(\pi) + Q_2((\theta_k)_{k\in\{1,\dots,K\}}).\end{aligned}$$

with $\tau_{t,k} = p(z_t = k|x_t, \theta)$ the membership posterior distribution estimated in (3) during the expectation step, and $f_{\theta_k,t}$ the density associated to cluster k regression model at time t. This optimization can therefore be achieved by the separate maximization of Q_1 and Q_2. The optimization of Q_2 with respect to the parameters $\theta_k = (\beta_k, \sigma_k)$ provides the following expressions:

$$\tilde{\beta}_k = arg \min_{\beta_k} \sum_{t=1}^{T} \tau_{t,k}(x_t - \sum_{r=1}^{R} \beta_k \phi_r(t))^2, \tag{5}$$

$$\tilde{\sigma}_k^2 = \frac{1}{\sum_{t=1}^{T} \tau_{t,k}} \sum_{t=1}^{T} \tau_{t,k}(x_t - \tilde{\mu}_k(t))^2, \tag{6}$$

with $\tilde{\mu}_k(t) = \sum_{s=1}^{S} \tilde{\beta}_{k,s}\phi_s(t)$ the estimated value of x_t by the regression model of cluster k.

4.2 Adaptive model selection strategy

In the initial model [20], the regression polynomial basis is common to every component, while in our contribution each regression order is specific. Moreover, we do not make *a priori* assumptions on the segment's number, which is also estimated by our strategy. To estimate both the segment's number and the polynomial regression order on each segment, we combine this model with an innovative top-down strategy. The strategy is iterative and consists, at each step, in identifying the 'worst' component, in terms of the partial likelihood defined as:

$$l_k(x;\theta) = \frac{1}{\sum_{t=1}^{T} \pi_{t,k}} \sum_{t=1}^{T} \pi_{t,k} log\left(f_{\theta_k}(x_t)\right), k \in 1,..,K.$$

This criterion quantifies a component representation quality weighted by the conditional membership probabilities. By improving the component $k_{old} \in \{1, \ldots, K\}$ that minimizes this score, two candidate models are created and compared. Splitting k_{old} in two sub-components, while conserving the other components, produce the first candidate model. We denote by k_1 and k_2 these new clusters. We denote t_m the weighted median of the sequence $\{1, \ldots, T\}$ with weights $\pi_{k_{old}}$, and consider this time as the optimal cut-point for splitting the component $\pi_{k_{old}}$. The observations membership probabilities associated are based on the former component membership probabilities. The component k_1 membership probabilities are defined as follows:

$$\pi_{k_1} = \begin{cases} \pi_{t,k_{old}} & , t \in \{1, \ldots, t_m\} \\ \epsilon & , t \in \{t_m + 1, \ldots, T\} \end{cases} , \tag{7}$$

with ϵ the threshold precision. The new cluster k_2 membership probabilities are obtained likewise, with inverted time indices. A regularization of the $(\pi_k)_K$ is necessary at this point to enforce the constraint $\sum_{k=1} \pi_{k,t} = 1, \forall t \in \{1, \ldots, T\}$. Increasing the order of k_{old} component regression model by one produces the second candidate. Two runs of EM are then launched, each of them considering one of the candidates as the initial state. After the convergence of both EM, the candidate optimizing the Bayesian Information criterion [22] is selected for the next iteration. This criterion is a score penalizing the likelihood of the model by its complexity. Given a model M with parameters Θ of size C, a set of observation x of size n, the BIC is defined as follows:

$$BIC(X, M, \Theta) = C \ln(n) - 2 \ln(L(X, M, \Theta)). \tag{8}$$

This strategy is summarized in algorithm 1.

Algorithm 1: Top down segmentation strategy.

```
Fix the convergence threshold $c > 0$
Choose an initial state for the first EM run:
$\theta_{old}^{init} := ((w_k, \beta_k, \sigma_k^2)_{k \in \{1,\ldots,K\}})_{old}^{init}$
Compute $\pi_{old}^{init}$ using equation (2)
Estimate $\theta_{old}^{end}$ by applying the EM algorithm
while relative increment in BIC $> c$ do
    Construct the first candidate model $\theta_{addSeg}^{init}$ with equation (7)
    Estimate $\theta_{addSeg}^{end}$ by applying the EM algorithm
    Construct the second candidate model $\theta_{incDeg}^{init}$ by increasing the
      least efficient component of the former mixture by one.
    Estimate $\theta_{incDeg}^{end}$ by applying the EM algorithm
    $\theta_{old}^{end} = \arg\max_{\theta \in \{\theta_{addSeg}^{end}, \theta_{incDeg}^{end}\}} BIC(\theta)$
end
```

After convergence of BIC criterion, we estimate the moments of regime change by choosing the maximum of membership probabilities. In Figure 2, we show the result of segmentation over a few time series from our use case AEB. This segmentation method is applied individually to each time-series and transforms each one in a set of sub-sequences. A segmentation result of an AEB time series is shown in Figure 2.

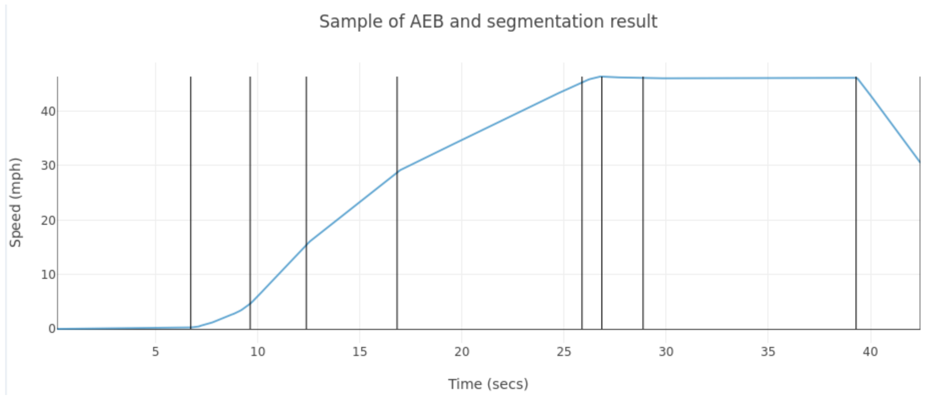

Fig. 2. Segmentation result sample.

4.3 Dictionary construction

Expressing the extracted segment in a common basis is mandatory to compare and cluster the sequences. This common basis, or dictionary, is constructed with clustering algorithm applied to the dataset composed of all segments from time series. The objective is to encode the original time-series in the new dictionary, as represented in Figure 3. The sub-segments are first scaled, expressed on

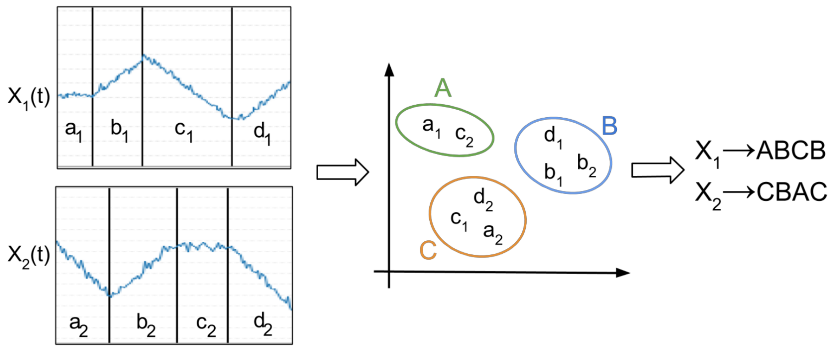

Fig. 3. *SDLHC*: From time-series to categorical sequences.

a common support, and regressed in a polynomial regression basis. Other informative descriptors can be added depending on the case, as the regime's duration, offset, or variance. These features are then clustered with a Gaussian mixture model (GMM) to produce the dictionary. In section 4.1, we mentioned an implicit assumption based on the segmentation polynomial basis. In this section we make the additional implicit assumption that the GMM is adapted to the regimes density estimation and makes sense from the field expert point of view.

The EM algorithm is initialized with the K-means++ algorithm, which is a standard approach [2]. At the end of this step, the modes of the Gaussian mixture components are the reference regimes, entitled "patterns" in the following, with which to recode the original time series. The dictionary size is determined by the field experts, assisted by the BIC.

An example of a dictionary with five patterns is shown in Figure 4. Using this dictionary, Figure 5 shows the encoded sequence. Two stationary patterns can be recognized (b and c), corresponding to cruise speed phases, as well as two accelerating (d and e) and one decelerating (a).

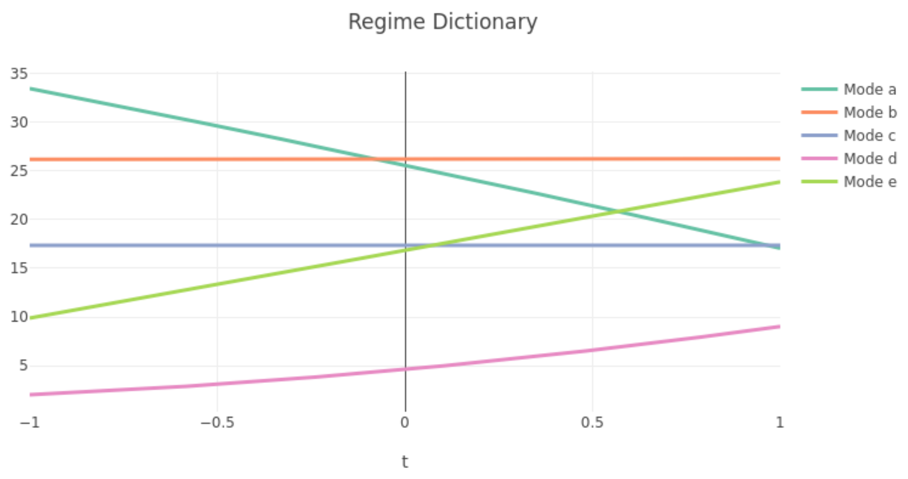

Fig. 4. Dictionary produced in the AEB use case.

After this re-coding phase, data dimension is greatly reduced: for a time-series of size n, the dimension goes from $\mathbb{R}^n$ to D^k, with D the categorical space and k the number of regimes composing the sequences. Clustering these sequences is $SDLHC$ third step subject.

4.4 Categorical Sequences Clustering

We use the Levenshtein distance [14] combined with Ward's hierarchical clustering method to obtain the final clusters. Levenshtein distance between

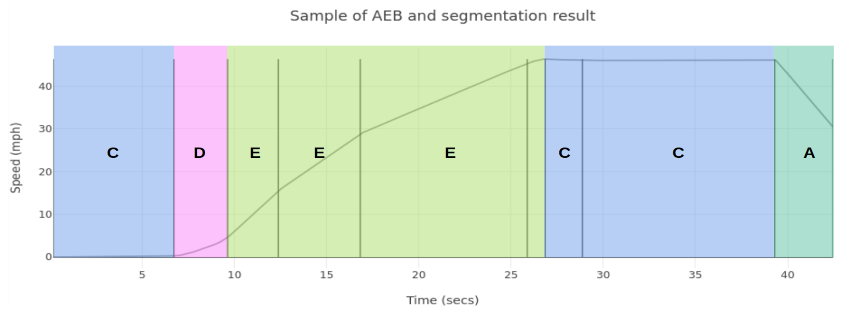

Fig. 5. Segment sequence encoded using the dictionary. Two stationary patterns can be recognized (b and c), corresponding to cruise speed phases, as well as two accelerating (d and e) and one decelerating (a).

two categorical sequences a and b (of size s_a and s_b) is defined as the minimum number of operations (insertion, deletion, substitution) needed to transform a into b. In this categorical space, Levenshtein Distance complexity is $O(s_a \times s_b)$. In the original Levenshtein distance, replacing a symbol with another has a fixed unit cost, independently from the target and replacement symbols. Therefore, it does not account for the fact that the two patterns may be more or less close. Figure 4 shows that some speed patterns are similar (different phases of acceleration of cruising speed) and distance on categorical sequences should take this similarity into account. The proposed Weighted Levenshtein Distance allows integrating this information into our clustering. Considering a set of patterns $R = \{r_s\}_S$, the edition cost between $r_a, r_b \in R$ is symmetric and defined as follows:

$$C(r_1, r_2) = \frac{||r_1 - r_2||_p}{\max_{s,t} ||r_s - r_t||_p}, \tag{9}$$

where $||.||_p$ is the p norm on the pattern space. The choice of p influence moderately the final clustering. In our AEB use case, experience led to the choice $p = \infty$. During the second phase of $SDLHC$, the dictionary is constructed based on scaled segments of same support, with optional addition of offset, variance or phase duration. During this part of categorical sequence clustering, the same features can be integrated to the edit operation cost computation.

Once the weighted Levenshtein Distance Matrix computed, Ward's hierarchical clustering method is applied to produce the final clusters. We compare the results with those of other state-of-the-art methods to prove the method capacity to produce a clustering with good performance.

5 Experiments

We present, in this section, the results of several experiments on public datasets and on a real-world use case AEB obtained from Renault's simulation system. The method described in this article was implemented in Scala for the segmentation step and R for the hierarchical step. Code and (public) datasets available at https://github.com/sdlhc-01/SDLHC.

The following baseline methods are selected:

- Three methods based on classical measures (Euclidean distance and DTW) associated with Partitional Around Medoid (PAM) clustering approach. We have also tested the combination of DTW with center construction using the popular Dynamic Barycenter Averaging (DBA) method [17].
- The $K - Shape$ method [16], a partitional clustering using the shape-based distance based on the cross-correlation measure.
- The SAX method, a dictionary-based methods from [15] that builds representations of the time series based on uniform time step segmentation. Based on this representation and associated distance, hierarchical clustering with Ward's criterion produce the clusters.
- In order to compare to the original method we aimed to extend, the results of $mixRHLP$ are also reproduced here.

Whenever baseline methods require it, we interpolate time-series to equal-length sequences. We used the R package *TSclust*'s distance-based and SAX methods implementations and $mixRHLP$ using *flamingos* R package. Some of these methods depend on parameters, usually estimated by optimizing a risk in a supervised framework. The comparison is based on the Adjusted Rand Index (ARI), a popular score in the clustering validation context. This criterion represents the proportion of correctly grouped and separated observations with respect to the observed classes. The ARIs obtained here are always the maximal ARI obtained when testing the method on a parameter grid, displayed in Table 1, reproducing the results that experts can obtain after fine-tuning.

Table 1. Parameters grid for ARI evaluation.

Method	Parameters	Range
SAX	Number of segments	(5,10,20,30,40,50)
	Number of gaussian bins	(2,3,5,7,10,20,30,40,50)
SDLHC	Dictionary size	(2,...,12)
MIXRHLP	Number of segments	(1,...,10)
	Polynomial regression order	(1,...,3)

5.1 Public datasets results

In order to validate $SDLHC$ adequation to the regime-changing time series clustering problematic, we selected a subset of the UCR archive [5] whose data exhibit regime structure. The ARI score obtained are shown in Figure 2. Although performant when applied to Renault's dataset (c.f. next subsection), we found that the weighted Levenshtein hierarchical clustering requires fine-tuning to adapt to the considered data characteristics. The test ran in this section therefore use the non-weighted Levenshtein distance. The results confirm

Table 2. Adjusted Rand Index on the UCR archive datasets.

Name	L2.PAM	DTW.PAM	DTW.DBA	K-SHAPE	SAX	MIXRHLP	SDLHC
CBF	0.28	0.66	0.68	0.63	0.46	0.47	0.71
OliveOil	0.46	0.53	0.40	0.50	0.00	0.40	0.55
Trace	0.32	0.40	0.66	0.57	0.32	0.41	0.94

that the method perform well when addressing regime-changing time series. In these tests, the considered datasets contain equal-length time-series. However, $SDLHC$ can also be applied, without data preprocessing, to unequal-length time-series, which is the case in our application.

5.2 Real dataset results

In the following section, we evaluate the clustering performance of $SDLHC$ on an industrial use case: the Autonomous Emergency Braking (AEB) system validation. In this case, a ground truth is available, and it is possible to compare clustering methods based on the similarity between the observed labels and the produced clusters. The clustering methods performances are, as in the previous section, measured by the ARI score. Renault's dataset is composed of 150 time series, with a duration varying from 13 to 52 seconds and length varying from 415 to 573 data points. The scores are obtained in the same conditions than the previous tests on public datasets, displayed in Figure 6. Two versions of $SDLHC$

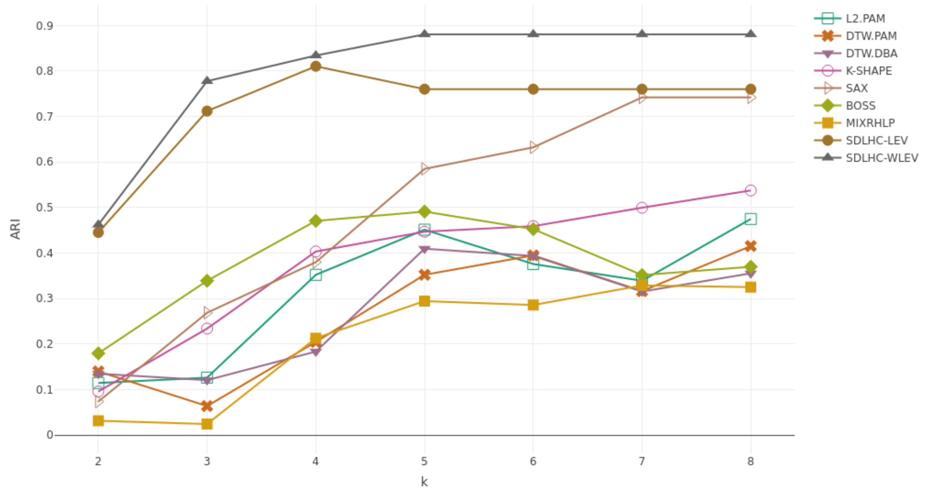

Fig. 6. ARIs scores of various clustering approaches as a function of the number of clusters.

are tested: $SDLHC-LEV$ and $SDLHC-WLEV$ corresponding to the use of the standard and weighted Levenshtein distance in $SDLHC$'s last step. ARI criterion confirms that the $SDLHC - WLEV$ method slightly improves the score obtained by $SDLHC - LEV$. Among the distance-based methods, the $K - Shapes$ method is the best performer without, however, reaching the ARI threshold of 0.45 regardless of the number of clusters. With high cluster numbers, SAX method nearly reaches the performance of $SDLHC - LEV$. This seems logical given the proximity between the proposed workflow and the dictionary-based methods.

6 Conclusions

In the context of unsupervised classification of regime-changing time-series, we propose a dictionary-based method that consists in three steps: automatic segmentation of each time-series, regime dictionary construction, and clustering of produced categorical sequences. $SDLHC$ shows good results when applied to time-series complying to the regime construction assumption, and is competitive with other state-of-the-art methods in this case. The ability to address unequal-length time-series, a-synchronized time-series, and time-series exhibiting asynchronous regimes are its best assets. The current assumptions on the polynomial regression basis for segmentation are adapted to experimental cases, but may not be suited to other physics-oriented use cases. In these circumstances, the Fourier polynomial basis may be another candidate to fit regimes and time-series. In this case, it is possible to re-interpolate the Fourier coefficient to compare regimes on a common basis, and even regimes from different sources, leading to the possibility of multivariate clustering. In this context, our current investigations are focusing on model selection and reduction through co-clustering.

Acknowledgements

The authors want to thank Groupe Renault and the ANRT (french National Association for Research and Technology) for their financial support via the CIFRE convention n° 2018/1114.

References

1. Batista, G.E., Wang, X., Keogh, E.J.: A complexity-invariant distance measure for time series. In: Proceedings of the 2011 SIAM international conference on data mining. pp. 699–710. SIAM (2011)
2. Blömer, J., Bujna, K.: Simple methods for initializing the em algorithm for gaussian mixture models. CoRR (2013)
3. Chamroukhi, F., Samé, A., Govaert, G., Aknin, P.: Time series modeling by a regression approach based on a latent process. Neural Networks **22**(5-6), 593–602 (2009)

4. Chamroukhi, F., Samé, A., Govaert, G., Aknin, P.: A hidden process regression model for functional data description. application to curve discrimination. Neurocomputing **73**(7-9), 1210–1221 (2010)
5. Dau, H.A., Keogh, E., Kamgar, K., Yeh, C.C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The ucr time series classification archive (October 2018), `www.cs.ucr.edu/~eamonn/time_series_data_2018/`
6. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society: Series B (Methodological) **39**(1), 1–22 (1977)
7. Deng, H., Runger, G., Tuv, E., Vladimir, M.: A time series forest for classification and feature extraction. Information Sciences **239**, 142–153 (2013)
8. Fuchs, E., Gruber, T., Nitschke, J., Sick, B.: Online segmentation of time series based on polynomial least-squares approximations. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(12), 2232–2245 (2010)
9. Kehagias, A.: A hidden markov model segmentation procedure for hydrological and environmental time series. Stochastic Environmental Research and Risk Assessment **18**(2), 117–130 (2004)
10. Keogh, E., Chu, S., Hart, D., Pazzani, M.: Segmenting time series: A survey and novel approach. In: Data mining in time series databases, pp. 1–21. World Scientific (2004)
11. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowledge and information systems **7**(3), 358–386 (2005)
12. Lavielle, M., Moulines, E.: Least-squares estimation of an unknown number of shifts in a time series. Journal of time series analysis **21**(1), 33–59 (2000)
13. Lee, W.H., Ortiz, J., Ko, B., Lee, R.: Time series segmentation through automatic feature learning. arXiv preprint arXiv:1801.05394 (2018)
14. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet physics doklady. vol. 10, pp. 707–710 (1966)
15. Lin, J., Keogh, E., Lonardi, S., Chiu, B.: 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. pp. 2–11 (2003)
16. Paparrizos, J., Gravano, L.: k-shape: Efficient and accurate clustering of time series. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. pp. 1855–1870 (2015)
17. Petitjean, F., Ketterlin, A., Gançarski, P.: A global averaging method for dynamic time warping, with applications to clustering. Pattern Recognition **44**(3), 678–693 (2011)
18. Sakoe, H.: Dynamic-programming approach to continuous speech recognition. In: 1971 Proc. the International Congress of Acoustics, Budapest (1971)
19. Salvador, S., Chan, P.: Toward accurate dynamic time warping in linear time and space. Intelligent Data Analysis **11**(5), 561–580 (2007)
20. Samé, A., Chamroukhi, F., Govaert, G., Aknin, P.: Model-based clustering and segmentation of time series with changes in regime. Advances in Data Analysis and Classification **5**, 301–321 (2011), `https://chamroukhi.com/papers/adac-2011.pdf`
21. Schäfer, P.: The boss is concerned with time series classification in the presence of noise. Data Mining and Knowledge Discovery **29**(6), 1505–1530 (2015)
22. Schwarz, G., et al.: Estimating the dimension of a model. The annals of statistics **6**(2), 461–464 (1978)