Published article: https://doi.org/10.1016/j.engappai.2021.104446

Inertial load classification of low-cost electro-mechanical systems under dataset shift with fast end of line testing

Nicholas Valceschini[a], Mirko Mazzoleni[a], Fabio Previdi[a]

[a] *Department of Management, Information and Production Engineering, University of Bergamo, via Marconi 5, 24044 Dalmine (BG), Italy.*

Abstract

This paper presents a rationale for designing a machine learning algorithm under dataset shift. In particular, we focus on the classification of the inertial load of low-cost Electro-Mechanical Actuators (EMAs) into several weight categories. In these low-cost settings, due to uncertainties in the manufacturing process, raw materials and usage, even if the EMA part number is the same, its serial numbers (i.e. items or exemplars) may show different physical behaviors. Thus, a learning model trained on data from a set of items can perform poorly when applied to other ones. The proposed solution comprises tailored normalization and cross validation procedures for training the classifier, along with suitable End Of Line (EOL) experiments for the characterization of a new produced EMA item. The approach is experimentally validated on the classification of the mass of sliding gates, using only measurements available on the gate EMA.

Keywords: Classification, dataset shift, EMA, end of line testing

Acronyms

BEMF Back-ElectroMotive Force

CV Cross Validation

DC Direct Current

ECU Electronic Control Unit

EMA Electro-Mechanical Actuator

EOL End Of Line

GBT Gradient Tree Boosting

KNN K-Nearest Neighbour

LDA Linear Discriminant Analysis

ON Object-wise Normalization

OCV Object-wise Cross Validation

P/N Part Number

RMS Root Mean Square

SVM Support Vectors Machines

Email addresses: nicholas.valceschini@unibg.it (Nicholas Valceschini), mirko.mazzoleni@unibg.it (Mirko Mazzoleni), fabio.previdi@unibg.it (Fabio Previdi)

1. Introduction

1.1. Motivation

The advent of the "industry 4.0" paradigm has led, in recent years, to a progressive shift both in companies mentality and actions (Lamnabhi-Lagarrigue et al., 2017). The presence of pervasive sensing and seamless connectivity is transforming the amount of information that industries can store and manage (Baur and Wee, 2015). Businesses that are able to

leverage their data, in order to provide ad-hoc services to their customers, are going to gain a significant advantage over competitors who will not be ready on this front.

The focus of the present work is fostered by these ideas. Specifically, we consider the problem of *designing a machine learning model* based on measurements from *low-cost* Electro-Mechanical Actuators (EMAs).

Traditional EMA systems are composed by (i) motor, (ii) transmission and (iii) load (usually unknown). The applications of learning algorithms to EMAs cover a variety of very practical purposes, such as fault diagnosis (Mazzoleni et al., 2019) and friction estimation (Angeloni et al., 2015).

Fault diagnosis and condition monitoring are especially important in safety-critical and high-cost applications, such as the aerospace industry (Mazzoleni et al., 2021), where the newly produced EMA undergoes extensive End Of Line (EOL) testing to assure the required product quality, which require considerable investments in terms of costs, time and need for expertise from the EOL operators. In the case of *low-cost applications*, an extensive EOL experimental campaign for evaluating or tuning the final product is not feasible, and it is often not performed at all. This, combined with the overall lower quality of the raw materials and variations in the EMA deployment environments, brings to a final product that has wide variations in its physical parameters and behaviour.

In this setting, training a learning algorithm (Theodoridis, 2020; Strang, 2019) may be difficult since the features extracted from the different items can have *different distributions*. This problem is generally known as *dataset shift* in the machine learning community (Moreno-Torres et al., 2012). There exists different cases of dataset shift, depending on the quantities that differ between train and test data. Let $P_{\text{test}}(\mathbf{x})$ be the distribution of the features vector $\mathbf{x}$, and $P_{\text{test}}(y|\mathbf{x})$ the conditional distribution of the output y given $\mathbf{x}$ in the test data. The quantities $P_{\text{train}}(\mathbf{x})$ and $P_{\text{train}}(y|\mathbf{x})$ are defined similarly for train data. Then, we can distinguish between:

- *covariate shift*, if $P_{\text{test}}(\mathbf{x}) \neq P_{\text{train}}(\mathbf{x})$ but $P_{\text{test}}(y|\mathbf{x}) = P_{\text{train}}(y|\mathbf{x})$. A recent and widespread technique that deals with this problem is the batch normalization in training deep neural networks (Ioffe and Szegedy, 2015);
- *prior probability shift*, if $P_{\text{test}}(y) \neq P_{\text{train}}(y)$ but $P_{\text{test}}(y|\mathbf{x}) = P_{\text{train}}(y|\mathbf{x})$;
- *concept shift*, if $P_{\text{test}}(\mathbf{x}) = P_{\text{train}}(\mathbf{x})$ but $P_{\text{test}}(y|\mathbf{x}) \neq P_{\text{train}}(y|\mathbf{x})$;
- *general dataset shift*, if $P_{\text{test}}(\mathbf{x}, y) \neq P_{\text{train}}(\mathbf{x}, y)$ but none of the above hold.

The main causes for dataset shift are *sample selection bias* and *non-stationary environments*, see (Moreno-Torres et al., 2012). While literature focused mostly on covariance, concept and prior probability shifts, the case of *general dataset shift* has received less attention.

Several methods were developed for detecting if the data suffer from general dataset shift, and tackled it with domain adaptation and data weighting methods (Yang et al., 2008; Raeder and Chawla, 2009). Since access to deployment environment data may not be available during training, it is not always feasible to employ those techniques to directly optimize the model for the target domain. Authors in (Subbaswamy et al., 2019) proposed to incorporate prior assumptions on how the data distribution might change, in the form of a Direct Acyclic Graph and using do-calculus (Pearl, 2009). However, in our low-cost setting, this approach presents an inherent difficulty.

In this work, we specifically aim to develop a *classification* algorithm to assign the EMA inertial load into a set of *weight categories*, under *dataset shift*. The problem is solved by introducing a set of *EOL characterization experiments* that are as low demanding as possible. We restrict our analysis to a single EMA *Part Number* (P/N) (i.e. a single EMA design), and suppose that different *serial numbers* or *items* of the same P/N (i.e. specific physical instantiations of one P/N) are available for the analysis. These working hypothesis do not compromise the applicability of our method, that can be straightforwardly applied to other part numbers and other types of learning problems than load classification.

1.2. Applicative context

The proposed learning procedure is applied in the context of EMAs designed for the actuation of heavy sliding gates. Legislative regulations, see e.g. (European Committee for Standardization CEN/TC 33, UNI EN 12453:2017, 2017), restrict the maximum reachable speed of the gate *depending on its weight range* for preventing safety issues and accidents. These ranges are defined in terms of weight categories, e.g. $[0-100), [100-200), [200-300), [300-400), [500-600)$ kg. Lighter gates are allowed to reach higher opening and closing speeds than heavier ones. In the current practice, the EMA item is configured for working at a specific maximum speed (default speed) with a fast EOL configuration.

When an actuator is mounted on a gate whose mass is too high, the maximum allowed motor speed have to be lowered by a configuration procedure. However, since the gate mass is usually not known, the configuration performed by the technician can only be of a qualitative type. In these cases, due to the lack of information about the gate mass, an EMA which maximum speed is way lower than what allowable for the mass of the gate is often adopted to be sure to meet the regulations, leading to a reduced user experience (since the gate moves slower than it could be). A system able to estimate the gate mass category will be beneficial for the actuator company from various points of view, such as (i) improved user experience; (ii) less P/N diversity; (iii) money savings.

The cost-effectiveness of a machine learning solution for such low-cost products may be seen from different perspectives. First, we remark that the cost effort is mainly faced during the design phase of the algorithm. Once the algorithm has been frozen, it can be implemented on all the EMA electronic control units without additional overheads, if not for the engineering costs. Second, we notice that the proposed approach may be empowered also in other situations where dataset shift problems are present. Finally, the approach may be cost-effective when considering that the tuning of a newly installed EMA (e.g. because the gate mass is unknown and the EMA power have to be regulated accordingly) entails physically sending a specialized operator in the installation place, with related costs. So, an automatic approach might reduce these expenses.

Summarizing, the aim of this work is to classify the mass of the gate into four weight categories {`Light`, `Medium`, `Medium-Heavy`, `Heavy`} using *only motor measurements* from the low-cost EMA, i.e. current and speed. More specifically, the contributions of this paper are:

1. we propose a procedure for designing a classifier under dataset shift that:
   - does not require prior knowledge from the user on how the data distribution might change;
   - does not need access to deployment environment data.

   The algorithm should have low computational requirements, since it must run on the EMA Electronic Control Unit (ECU). In this phase, we developed tailored normalization and cross-validation procedures, that we called respectively Object-wise Normalization (ON) and Object-wise Cross Validation (OCV), that take into account that the training data come from different serial numbers (items) of the same P/N;
2. we propose a practical method for characterizing new produced EMA items by fast and affordable EOL experiments, that do not impact on the costs of the company that employs the low-cost EMAs;
3. we evaluate the approach by employing a real sliding gate. The most important features are selected by feature importance and selection steps, in order to simplify the computational burden of the algorithm. The trained classifier is evaluated on experimental data using two different environments (internal and external ambient).

The remainder of the paper is organized as follows. Section 2 presents the full procedure for training and evaluate a classifier under dataset shift, considering also the characterization of new EMA items. Section 3 applies the proposed procedure to an experimental application, describing the experimental setup, tests protocol, and the rationale for choosing the final classifier. Section 4 evaluates the classification results

under different points of views, including cross validation performance, required number of EOL tests for EMA items characterization and simulation of a real-world application of the procedure on test data. Lastly, Section 5 is devoted to concluding remarks and future developments.

2. Classifier design procedure for electromechanical actuators under dataset shift

This section describes the proposed procedure for training the classifier under dataset shift. The steps of the procedure are shown in Figure 1:

1. *data acquisition*: current and speed raw data are measured from the EMA item;
2. *feature extraction*: a set of d ad-hoc features are extracted from the raw data;
3. *feature normalization*: the features are normalized with a tailored normalization procedure, that we named Object-wise Normalization (ON);
4. *feature selection*: the $q \leq d$ most important features for the classification task are selected with an automatic procedure that relies on a tailored cross-validation scheme, that we named Object-wise Cross Validation (OCV);
5. *best algorithm selection*: a set of classification algorithms are compared, and the best one is selected as a trade-off between performance (considering the OCV index) and time required to perform a prediction.

2.1. Object-wise normalization

Suppose to have at disposal a training set $\mathcal{D} = \{\mathbf{x}_j, y_j\}_{j=1}^{N}$, composed by N features vectors $\mathbf{x}_j \in \mathbb{R}^{d\times 1}$, where y denotes the (categorical) target variable. These N observations come from a set of M items $\mathcal{M} = \{m_1, m_2, \ldots, m_M\}$ of the same EMA P/N. These items will be used for training the classification algorithm.

Given all the values of the h-th feature $\mathbf{x}^{[h]} \in \mathbb{R}^{N\times 1}$, their normalized value $\tilde{\mathbf{x}}^{[h]}$ can be computed as (Jayalakshmi and Santhakumaran, 2011):

$$\tilde{\mathbf{x}}^{[h]} = \frac{\mathbf{x}^{[h]} - \text{mean}\left\{\mathbf{x}^{[h]}\right\}}{\text{std}\left\{\mathbf{x}^{[h]}\right\}}, \quad h = 1, ..., d, \qquad (1)$$

where $\text{mean}\left\{\mathbf{x}^{[h]}\right\}$ and $\text{std}\left[\mathbf{x}^{[h]}\right]$ denote the mean and standard deviation of the $h-$th feature $\mathbf{x}^{[h]}$, respectively. The normalization procedure in (1) considers *all* the N *data*, i.e. the features extracted by considering all M items, for the computation of $\text{mean}\left\{\mathbf{x}^{[h]}\right\}$ and $\text{std}\left\{\mathbf{x}^{[h]}\right\}$, so it *does not account for the peculiar characteristics of the different items*, i.e. the fact that the average and standard deviation values of each feature vary from one item to another, not only due to measurements noise, but also because of the dataset shift problem. Standard normalization in (1) assumes that the features vector come from the same distribution, but, in our setting, this is not the case. A better option would be to normalize each training item in $\mathcal{M}$ by *its own* features averages and standard deviations. Thus, we propose the *Object-wise Normalization* (ON) procedure that explicitly takes into account the peculiar characteristics of each item data. It is composed by the following steps, see Algorithm 1:

1. split temporarily the training set $\mathcal{D}$ by item, such that each sub-dataset $\mathcal{D}_s$, $s = 1, \ldots, M$, $\bigcup_s \mathcal{D}_s = \mathcal{D}$, consists of the N_s observations computed from item m_s, with $\sum_{s=1}^{M} N_s = N$;
2. perform standard normalization (1) on each $\mathcal{D}_s$, obtaining the normalized sub-datasets $\tilde{\mathcal{D}}_s$;
3. merge all normalized sub-datasets $\tilde{\mathcal{D}}_s$ in a new dataset $\tilde{\mathcal{D}}$, composed again by all N observations.

Algorithm 1: Object-wise normalization

Input: Raw dataset $\mathcal{D}$
Output: Normalized dataset $\tilde{\mathcal{D}}$
for $s = 1, \ldots M$ **do**
  Create the dataset $\mathcal{D}_s$ splitting $\mathcal{D}$ by item
  $\tilde{\mathcal{D}}_s \leftarrow$ Normalize $\mathcal{D}_s$ using (1)
return $\tilde{\mathcal{D}} = \bigcup_s \tilde{\mathcal{D}}_s$

2.2. Object-wise cross validation

Cross Validation (CV) is one of the most widely used methods for estimating the generalization error of a learning model (Hastie et al., 2009, Chapter 7).

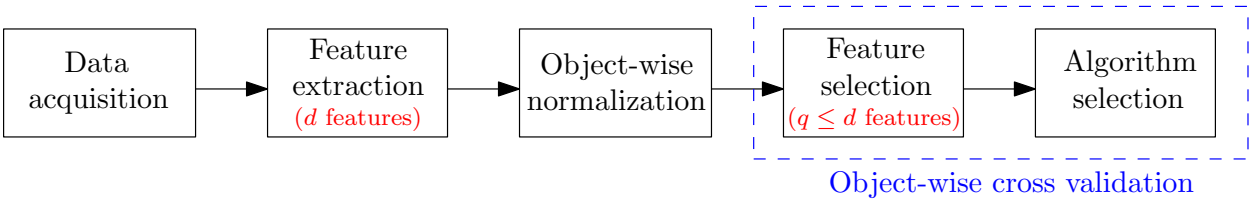

Figure 1: Procedure for the design of the classification algorithm for EMA items.

K-fold cross validation splits the data into K parts (folds). Iteratively, the model is trained on $K-1$ parts and evaluated on the held-out part that is not used for training.

In the considered application, the aim is to assign the inertial load actuated by the EMA item to a mass category, relying only on its data. When CV is used for model selection and assessment, the held-out fold (used for validation purposes) consists in a mix of data from *all* the employed training items in $\mathcal{M}$. Thus, this process does not correctly mimic a real-world application scenario, where the classifier is required to provide its classification based only on the data from a *single* item.

To solve this issue, we propose a variation of the CV rationale that we named *Object-wise Cross Validation* (OCV). The main difference with the standard CV is that the OCV composes the folds with *only the data from a specific item*, see Figure 2. Formally, let $\delta : \{1, \dots, N\} \to \{1, \dots, M\}$ be an indexing function that indicates the EMA item that generated the j-th observation. Then, the OCV error is computed as

$$\mathrm{OCV}(\hat{f}, \boldsymbol{\theta}) = \frac{1}{N} \sum_{j=1}^{N} \ell \left(y_j, \hat{f}^{-\delta(j)} \left(\mathbf{x}_j, \boldsymbol{\theta} \right) \right), \qquad (2)$$

where the k-th fold contains data from the k-th EMA item, and in this case $K = M$, so that $k = 1, \dots M$.

In the following, the OCV method will be applied two times:

1. for a *model selection* purpose, see (Arlot et al., 2010), by estimating the features importance in order to choose a reduced set of $q \leq d$ features to design the classifier. This step is performed by using a Gradient Tree Boosting (GTB) model (Friedman, 2001);
2. for a *model assessment* purpose, to compare between A different classification algorithms that rely on the q selected features.

Remark 1. *In a standard machine learning problem, i.e. where the train and test data come from the same distribution, the CV procedure would require to normalize (with standard normalization) the held-out fold with the statistics computed on the train folds, for each iteration of the CV procedure. Here, this strategy does not apply since the data of different folds do not share the same properties, see Section 2.1. So, each data fold is normalized with its own normalization quantities following the ON procedure.*

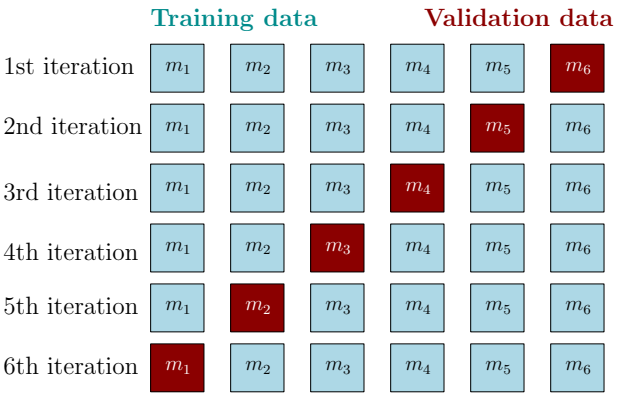

Figure 2: Example of fold extraction using the proposed object-wise cross-validation with six items.

2.3. Feature selection

A feature selection step is here employed to simplify the classifier computational requirements, leveraging the GTB algorithm (Pan et al., 2009). The

GTB generates a partition of the feature space into multiple regions, each one corresponding to one class. This partition is the result of a combination of decision trees, used as a base learner in the boosting procedure. The *overall importance* of each feature in solving the multi-class classification problem can be evaluated by combining the importance of that feature for all the trees, see (Hastie et al., 2009, Chapter 10).

The computed features importance allows to define a *feature ranking* $\mathscr{R}$, i.e. an ordered list from the most important feature to the least one. A baseline strategy for feature selection is to select the most important $q \leq d$ features in $\mathscr{R}$. However, the importance of these features is computed by considering also their *interaction with all the other features*. So, it is not guaranteed that the best model with q features (in terms of generalization capability) will be composed by the q most important ones, see Pan et al. (2009).

The procedure in Algorithm 2 is proposed to select the best set of q features, starting from the normalized dataset $\tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_j, y_j\}_{j=1}^N$:

1. Compute the features importance using the OCV procedure of Section 2.2. This produces a feature ranking $\mathscr{R}_k$ for each of the $k = 1, \ldots, M$ cross validation folds;
2. for each feature, average its importance over the M folds, producing a final feature ranking $\mathscr{R}$;
3. select the best number of features applying Algorithm 3, where the first q features that induce an OCV performance greater than a performance threshold ε are selected.

In the following, we will denote with $\mathcal{Z} = \{\mathbf{z}_j, y_j\}_{j=1}^N$ the train dataset with $\mathbf{z}_j \in \mathbb{R}^{q\times 1}$ the j-th features vector, obtained from the raw features vector $\mathbf{x}_j \in \mathbb{R}^{d\times 1}$ using the selected q features. The normalized train dataset $\tilde{\mathcal{Z}} = \{\tilde{\mathbf{z}}_j, y_j\}_{j=1}^N$, is defined similarly using the normalized features vectors $\tilde{\mathbf{x}}_j$, and it will be used for *training* the classifier.

2.4. Best algorithm selection

The aim of this last step is to find the classification algorithm that has the best trade-off between classification performance and prediction time (i.e. the time for computing the model output). To this end, a number of A algorithms are compared, each one using the best q features selected using Algorithm 2 and the dataset $\tilde{\mathcal{Z}}$ (normalized with the ON procedure).

Computing the OCV for the A algorithms generates a set of A classification performances and A prediction times. So, we need to compose a unique ranking to select the best algorithm. A simple method is as follows, see Figure 3:

1. the algorithms are ranked by performance (decreasing order) and prediction time (increasing order);
2. apply a score to each algorithm in both rankings, such that the a-th algorithm gets a score $\alpha_1 = a$ for the performance ranking, and a score $\alpha_2 = a$ for the prediction time raking, with $a = 1, \ldots, A$;
3. obtain a single final score for each algorithm as $\rho = \alpha_1 + \alpha_2$;
4. create the final ranking by sorting with respect to the final score ρ in decreasing order. The first algorithm is chosen as the best one.

Remark 2. *The GTB algorithm of Section 2.3 is*

Algorithm 2: Feature importance

Input: Normalized dataset $\tilde{\mathcal{D}}$
Output: A feature ranking $\mathscr{R}$
Run OCV in (2) with GTB, computing the overall features importance for each $k = 1, \ldots, M$ fold, to obtain M feature rankings $\mathscr{R}_k$
Compute the ranking $\mathscr{R} = \frac{1}{M}\sum_{k=1}^{M} \mathscr{R}_k$
Select the best features using Algorithm 3

Algorithm 3: Feature subset selection

Input: Ranking $\mathscr{R}$ of d features, threshold ε
Output: A subset of $q \leq d$ features
for $q = 1, \ldots, d$ **do**
  Select the firsts q features of the ranking $\mathscr{R}$
  Run OCV in (2) with GTB using q features
  if *OCV performance* $> \varepsilon$ **then**
    return q
return d

employed only for a feature selection purpose, and it is not guaranteed to be the finally chosen classifier.

Remark 3. *The employed algorithm selection scheme is just one of the possible ways to select one classifier amongst several ones. The "optimality" of the selection scheme depends on what it is meant with "optimum": in our case, we want a trade-off between performance and computational time. With the used selection procedure, more than one algorithm may attain the same score on the ranking: in this case, the designer may select a classifier based on other criteria, such as its interpretability or availability of libraries for its implementation.*

2.5. EMA items characterization with end of line tests

The OCV requires that features extracted from an item are normalized with respect to their own average value and standard deviation. Many tests may be necessary to reliably estimate these normalization quantities. However, it is not reasonable to perform an high number of experiments at the end of the production line or before deploying the product in its environment, due to the low-cost of the considered EMA product and its non-critical applications.

Hence, we propose a different method to compute these normalization quantities by reducing, as much as possible, the number of needed EOL experiments to characterize a new item m^*. The rationale comprises the following steps, see Algorithm 4 and Algorithm 5:

1. for each $s = 1, \dots, M$ item in $\mathcal{M}$, compute a *prototype* point $\bar{\mathbf{z}}_s$ as

$$\bar{\mathbf{z}}_s = \left[\bar{z}_s^{[1]}, \dots, \bar{z}_s^{[q]}\right]^\top \in \mathbb{R}^{q\times 1}, \tag{3a}$$

$$\bar{z}_s^{[h]} = \text{mean}\left\{\mathbf{z}_s^{[h]}\right\}, \quad h = 1, \dots, q \tag{3b}$$

   with $\mathbf{z}_s^{[h]} \in \mathbb{R}^{N_s\times 1}$ containing the h-th feature values computed from the s-th item data in the not-normalized training set $\mathcal{Z}$, where N_s denotes the number of observations computed from the s-th training item in $\mathcal{M}$[1], with $N = N_s \cdot M$;
2. create a clustering $\mathscr{C}$ by grouping the M points $\bar{\mathbf{z}}_s$ into C clusters using a clustering heuristic (e.g. visual inspection of the points) or an automatic clustering procedure (Hastie et al., 2009, Chapter 13). Each cluster is characterized by its centroid $\mathbf{c}_\kappa \in \mathbb{R}^{q\times 1}, \kappa = 1, \dots, C$;
3. perform a reduced number of $N_s^* \ll N_s$ experiments on the new item m^*;
4. create the dataset $\mathcal{Z}^* = \left\{\mathbf{z}_j^*, y_j^*\right\}_{j=1}^{N_s^*}$ using data measured from m^*, where the features vector $\mathbf{z}_j^* \in \mathbb{R}^{q\times 1}$ follows from Section 2.3;
5. compute the prototype $\bar{\mathbf{z}}^*$ for item m^* as

$$\bar{\mathbf{z}}^* = \left[\bar{z}^{*[1]}, \bar{z}^{*[2]}, \dots, \bar{z}^{*[q]}\right]^\top \in \mathbb{R}^{q\times 1}, \tag{4a}$$

$$\bar{z}^{*[h]} = \text{mean}\left\{\mathbf{z}^{*[h]}\right\}, \quad h = 1, \dots, q, \tag{4b}$$

   where $\mathbf{z}^{*[h]} \in \mathbb{R}^{N_s^*\times 1}$ contains all the N_s^* values of the h-th feature;
6. find the nearest centroid $\mathbf{c}_{\kappa^*} \in \mathbb{R}^{q\times 1}$ to $\bar{\mathbf{z}}^*$;
7. assume that cluster of κ^* contains M^* items. For each of the $\xi = 1, \dots, M^*$ EMA items, get

$$\bar{\mathbf{z}}_\xi = \left[\bar{z}_\xi^{[1]}, \dots, \bar{z}_\xi^{[q]}\right]^\top, \text{ as in(3a)}$$

$$\breve{\mathbf{z}}_\xi = \left[\breve{z}_\xi^{[1]}, \dots, \breve{z}_\xi^{[q]}\right]^\top, \tag{5a}$$

$$\breve{z}_\xi^{[h]} = \text{std}\left\{\mathbf{z}_\xi^{[h]}\right\}, \quad h = 1, \dots, q \tag{5b}$$

8. the data in $\mathcal{Z}^*$ are then normalized with the quantities

$$\mathbf{m} = \text{mean}\left\{[\bar{\mathbf{z}}_1, \dots, \bar{\mathbf{z}}_{M^*}]\right\}, \tag{6a}$$

$$\mathbf{s} = \text{mean}\left\{[\breve{\mathbf{z}}_1, \dots, \breve{\mathbf{z}}_{M^*}]\right\}, \tag{6b}$$

   obtaining a normalized dataset $\tilde{\mathcal{Z}}^*$.

Summarizing, the proposed characterization approach normalizes the data $\mathcal{Z}^*$ of a new item m^* with the averaged means and standard deviations of the

[1] For simplicity, we consider N_s equal for all items.

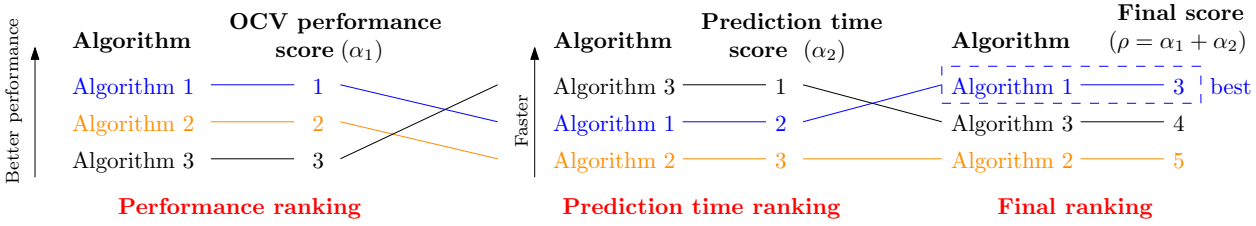

Figure 3: Rationale of the best algorithm selection.

training items in $\mathcal{M}$ that belong to the cluster κ^*, which centroid $\mathbf{c}_{\kappa^*}$ is nearest to the prototype (4a) computed from experiments on m^*. In this way, we obtain a normalized dataset $\tilde{\mathcal{Z}}^*$.

Algorithm 4: Training EMA items clustering

Input: Dataset $\mathcal{Z}$ of training experiments, a number of clusters C
Output: a clustering $\mathscr{C}$ of prototypes from $\mathcal{Z}$
for $s = 1, \ldots, M$ **do**
  for $h = 1, \ldots, q$ **do**
    Compute the average $\bar{z}_s^{[h]}$ of all values $\mathbf{z}_s^{[h]}$ of the h-feature in $\mathcal{Z}$, see (3b)
  Define $\bar{\mathbf{z}}_s = \left[\bar{z}_s^{[1]}, \ldots, \bar{z}_s^{[q]}\right]^\top$ as in (3a)
Cluster the data $\{\bar{\mathbf{z}}_1, \ldots, \bar{\mathbf{z}}_M\}$ into a clustering $\mathscr{C}$ of C clusters

Algorithm 5: EMA item characterization

Input: Raw dataset $\mathcal{Z}^*$ of EOL experiments, a clustering $\mathscr{C}$
Output: Normalized dataset $\tilde{\mathcal{Z}}^*$
for $h = 1, \ldots, q$ **do**
  Compute the average $\bar{z}^{*[h]}$ of all values $\mathbf{z}^{*[h]}$ of the h-feature in $\mathcal{Z}^*$, see (4b)
Define $\bar{\mathbf{z}}^* = \left[\bar{z}^{*[1]}, \bar{z}^{*[2]}, \ldots, \bar{z}^{*[q]}\right]^\top$ as in (4a)
Assign $\bar{\mathbf{z}}^*$ to its nearest cluster κ^* in $\mathscr{C}$
$\tilde{\mathcal{Z}}^* \leftarrow$ normalize $\mathcal{Z}^*$ following (5) and (6)

3. Experimental application to the classification of sliding gates mass

This section applies the proposed classifier design procedure to the classification of the mass of sliding gates, actuated by the same EMA P/N for which we suppose to have at disposal several serial numbers (items) of it. The main parts consist of:

1. description of the experimental setup;
2. computation of the features from measured EMA item variables;
3. data normalization with the proposed ON procedure in Section 2.1;
4. selection of the best set of features as in Section 2.3 using the proposed OCV procedure in Section 2.2;
5. choice of the best classification algorithm as proposed in Section 2.4.
6. evaluation of the classification performance of the chosen algorithm on test data.

3.1. Data acquisition and experimental setup

This section describes: (i) the considered type of EMA P/N and the sliding gate employed; (ii) the available measurements with the experimental protocol for data acquisition.

3.1.1. Description of actuators and sliding gate

The experimental setup consists of a sliding gate actuated by a $V_0 = 24\,\mathrm{V}$ Direct Current (DC) motor. The mass of the gate can be increased by manually adding iron bars of specific weight to the gate structure, see Figure 4. We investigated four weight categories, i.e. {`Light`, `Medium`, `Medium-Heavy`, `Heavy`}

The categories correspond to the following gate mass ranges:

1. `Light`: $[250 - 350)$ kg;
2. `Medium`: $[350 - 450)$ kg;
3. `Medium-heavy`: $[450 - 550)$ kg;
4. `Heavy`: $[550 - 650)$ kg;

Experiments were performed with gates of about 300kg, 400kg, 500kg, and 600kg, so that all four categories are represented.

We consider $M = 6$ items of the same EMA P/N.

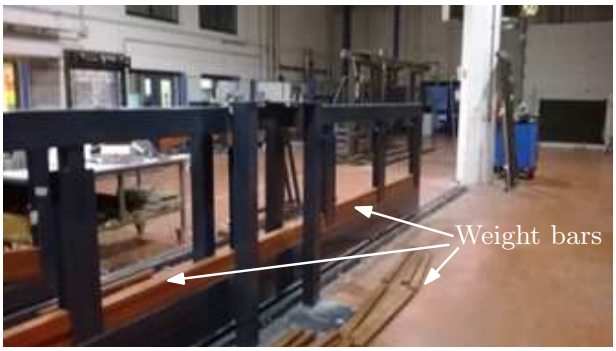

Figure 4: The sliding gate used for the experiments. The orange bars inside the gate are added and removed to change the overall gate mass.

The sliding gate moves by means of steel wheels on a steel rail. The motor-side rotation is transformed into a load-side rotation by an internal gear mechanism. The rotation at the output of the motor gear mechanism is transformed into linear motion by a pinion and rack transmission system, that connects the actuator to the gate, see Figure 5. The transmission ratio is $\tau_0 = \frac{\omega_L(t)}{\omega(t)} = \frac{1}{47}$, where $\omega_L(t)$ and $\omega(t)$ are the load and motor rotational speed, respectively, and $t \in \mathbb{Z}_{>0}$ denotes the t-th sampled time instant. The pinion consists of 44 teeth, with primitive radius of $r_0 = 28 \cdot 10^{-3}$ m. Considering the transmission as rigid, the rotational movement of the motor is then transmitted to the axial motion of the sliding gate by the transmission ratio $\bar{\tau}_0 = \tau_0 \cdot r_0 = 0.5957 \cdot 10^{-3}$ m. The actuator weights 11 kg and its dimensions are $249 \times 265 \times 203$ mm. The gate has dimensions of approximately $400 \times 150 \times 15$ cm.

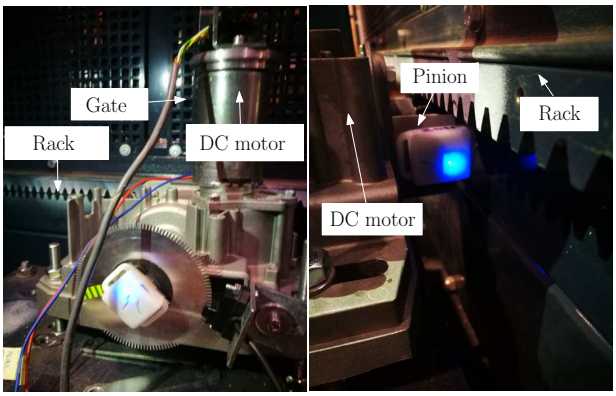

Figure 5: Example of an EMA item employed during the tests, highlighting the overall connection with the gate.

The motor and transmission parameters are summarized in Table 1. Due to the low-cost of the EMA P/N, the physical parameters have a variability of more than 20% on their nominal value.

Table 1: Nominal parameters of the considered EMA P/N.

Parameter	**Value**
Nominal voltage: V_0	24V DC
No load speed: ω_{M_1}	$3400 \pm 10\%$ rpm
No load current after 20s: i_1	1.4 A
Starting current: i_0	33 ± 1.5 A
Starting torque: T_0	2.2 ± 0.1 Nm
Rotor inertia: J_{m0}	$3.5 \cdot 10^{-5}$ kg $\cdot$ m^2
Gear ratio: τ_0	$\frac{1}{47}$
Primitive pinion radius: r_0	$28 \cdot 10^{-3}$ m
Transmission ratio: $\bar{\tau}_0$	$0.5957 \cdot 10^{-3}$ m
Torque constant: K_{T_0}	0.0685 Nm/A
Motor resistance: R_0	0.7473 Ω

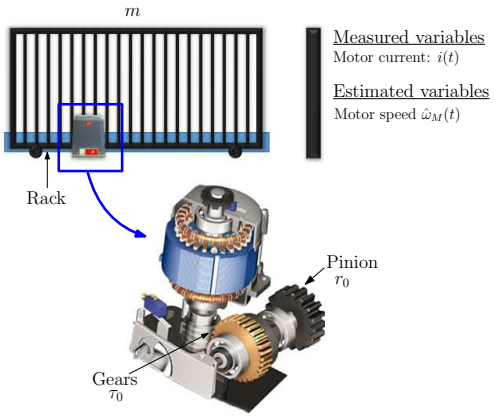

Figure 6: Schematic representation of the overall system. (Blue) Known parameters. (Red) Unknown parameters. (Green) Measured variables.

3.1.2. Experimental protocol

The EMA is controlled by a very simple electronics. This hardware allows the acquisition of several variables with a sampling frequency of $f_s = 100\,\text{Hz}$. The available measurements for each EMA item are (see Figure 6):

1. *motor speed estimate* $\hat{\omega}_M(t)$ computed from the Back-ElectroMotive Force (BEMF) values;
2. *motor current* $i(t)$, flowing in the motor coils;
3. *motor working phase* $p(t)$, showing which working phase the motor is, i.e. if it is in acceleration, constant or deceleration phase.

We remark here that these measurements are already available on the EMA, and the mass classification problem will be solved without any additional sensors.

The experiments consisted in performing open-loop experiments on the EMA items. Furthermore, we assume that the gate moves on a terrain with no slope. The input injected to the motor is a *trapezoidal voltage profile*, so we can distinguish three movement phases, measured in the variable $p(t)$:

1. acceleration phase;
2. constant velocity phase;
3. deceleration phase.

The rise and fall times of the acceleration and deceleration phases have been set to 1 s (the minimum settable acceleration/deceleration time). The rationale is to excite the system with the maximum possible dynamic (simulating a step response). The constant value of the input voltage trapezoidal profile (constant velocity phase) is set to 24 V, i.e. the nominal motor voltage.

Figure 7 depicts the experimentally measured $i(t), p(t)$ and $\hat{\omega}_M(t)$ data for $T = 20$ open-loop experiments (i.e. directly injecting a trapezoidal voltage profile input) on a single EMA item. It can be noticed how $i(t)$ increases and $\hat{\omega}_M(t)$ decreases as the weight grows.

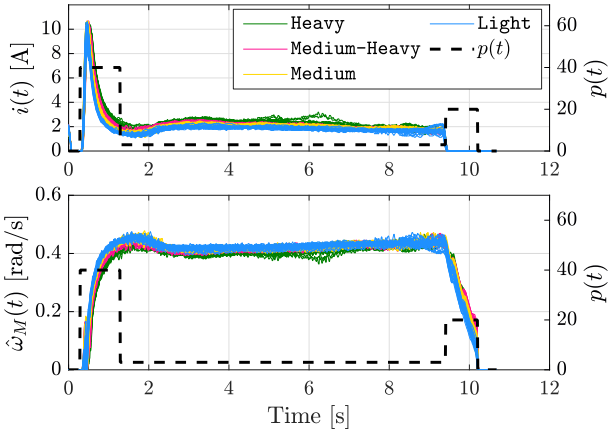

Figure 7: Current $i(t)$, estimated speed $\hat{\omega}(t)$ and phase $p(t)$ data from experiments with different weight classes on a single EMA item.

We can distinguish the following sources of variation that may affect the measured motors data:

- *motor physical properties*: different items of the same EMA, specifically the motor part. We considers $M = 6$ items $\mathcal{M} = \{m_1, m_2, \ldots, m_6\}$;
- *environment*: boundary experimental conditions such as gate binary, rack transmission and internal vs. external ambient. We consider $E = 2$ environments $\mathcal{E} = \{e_1, e_2\}$, in internal and in external ambient, respectively.

The experimental protocol consists in two setups aiming at investigating the possible sources of variation in the operative ambient of the EMA:

- **train setup:** the first type of experimental protocol consists in experiments on the EMAs in $\mathcal{M}$, varying the gate mass, in environment e_1, i.e. a gate binary and a rack in an internal ambient. We performed $T = 20$ experiments for each of the $W = 4$ weight classes and EMA items, for a total of $N_s = T \cdot W = 80$ tests per EMA item. These data are used to *train* the mass classification algorithm and define the *training set* $\mathcal{D}$.

- **test setup:** the second type of experiments considers only the *single* EMA item m_6 in the environment e_2, i.e. we changed the binary, the rack, the gate and the ambient (from an internal ambient to an external one). These data are used only to *test* the final algorithm, to evaluate its robustness at varying the environment and it will define the *test set* $\mathcal{D}_T$. This will be the typical situation of an industrialized application of the classifier, when the actuator is deployed on a customer sliding gate. Item m_6 has been randomly chosen to be used in environment e_2, and not by a specific or particular reason.

Summarizing, the total number of experiments that compose the training set is $N = T \cdot W \cdot M = N_s \cdot M = 480$ and the number of experiments that compose the test data is $N_T = T \cdot W \cdot 1 = 80$.

3.2. Feature extraction

The experimental dataset considered for training the mass classifier consists in the $N = 480$ curves of $\hat{\omega}_M(t)$, $i(t)$ and $p(t)$ measurements sampled in the internal environment e_1 at $f_s = 100\,\text{Hz}$, see Section 3.1.2. For each curve, we computed the following $d = 10$ features during *acceleration* and *constant velocity* movement phases:

- $r^{(a)}_{\hat{\omega}}$: Root Mean Square (RMS) of the estimated motor speed during the acceleration movement phase. Notice that this has the physical interpretation of being related to the kinetic energy of the EMA-gate system;

- $r^{(c)}_{\hat{\omega}}$: RMS of the estimated motor speed during the constant-speed movement phase;

- $r^{(a)}_{i}$: RMS of the motor current during the acceleration movement phase. This feature is related to the electric power of the actuator;

- $r^{(c)}_{i}$: RMS of the motor current during the constant-speed movement phase;

- $\bar{\hat{\omega}}^{(c)}$, $\max^{(c)}\{\hat{\omega}(t)\}$, $\min^{(c)}\{\hat{\omega}(t)\}$: average, max and min values of the estimated speed during constant-speed phase, respectively;

- $\bar{i}^{(c)}$, $\max^{(c)}\{i(t)\}$, $\min^{(c)}\{i(t)\}$: average, max and min values of the measured current during constant-speed phase, respectively.

Thus, the training set $\mathcal{D} = \{\mathbf{x}_j, y_j\}_{j=1}^{N}$ is composed by N feature vectors $\mathbf{x}_j \in \mathbb{R}^{d\times 1}$. The weight category labels are denoted with $y_j \in \{\texttt{Light}, \texttt{Medium}, \texttt{Medium-Heavy}, \texttt{Heavy}\}$. Thus, we have a $G = W = 4$ class classification problem.

Consider now, as an example case, the distribution of the feature $r^{(a)}_{\hat{\omega}}$ on data measured in environment e_1 (internal ambient). Figure 8 and Figure 9 depict experimental data acquired on the *single item* m_6[2], showing $r^{(a)}_{\hat{\omega}}$ variations by weight class and environment, respectively. In particular, Figure 9 highlights that the data distribution $P(r^{(a)}_{\hat{\omega}})$ is not stationary, i.e. *train and test data may have different distributions* due to changes in the environment, thus possibly degrading the predictive ability of the classification algorithm. Figure 10 shows the distribution of $r^{(a)}_{\hat{\omega}}$ between *all* the EMA items, considering all $N_s = 80$ tests (at the different $W = 4$ weights) for each item in environment e_1. It is possible to notice how the distribution of the data at different weight classes $P(r^{(a)}_{\hat{\omega}}|y)$ differs from one item to another. We assume that the items have an equal probability to be deployed on each gate weight category, so that $P(y = \texttt{Light}) = P(y = \texttt{Medium}) = \ldots = P(y = \texttt{Heavy})$, so that $P(y|r^{(a)}_{\hat{\omega}})$ differs from item to item.

[2]The behaviour of the other EMA items is similar.

Thus, we treat the weight classification problem as a *general dataset shift* one, see Section 1.

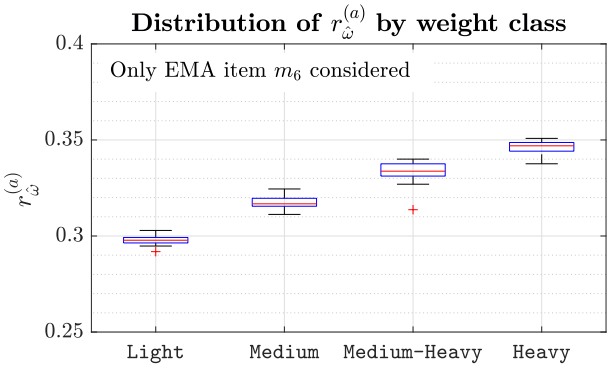

Figure 8: Distribution of the feature $r_{\hat{\omega}}^{(a)}$ at different weight classes on the single EMA item m_6 in environment e_1.

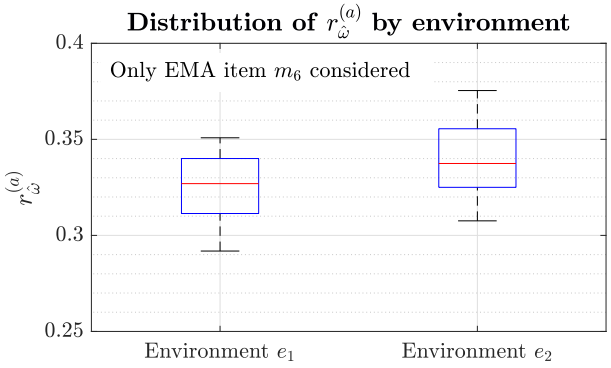

Figure 9: Distribution of the feature $r_{\hat{\omega}}^{(a)}$ at different environments on the single EMA item m_6.

3.3. Standard vs. object-wise normalization

It is interesting to compare the effect of the ON procedure, described in Section 2.1 and Algorithm 1, with respect to the standard normalization of (1) that employs the full dataset of N observation without discerning between their belonging to the different EMA items. This is assessed using the OCV procedure of Section 2.2. Since we have $M = 6$ items, the procedure generates M data folds $\mathcal{D}_s$, each one containing the data from item m_s, with $s = 1, \ldots, 6$.

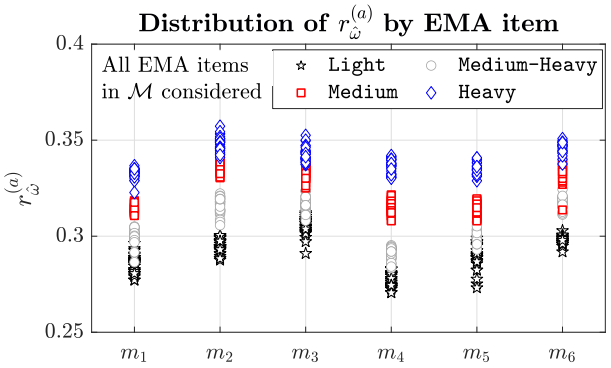

Figure 10: Distribution of the feature $r_{\hat{\omega}}^{(a)}$ at different EMA items in environment e_1 in different weight classes.

Table 2 reports the classification accuracy on each fold $\mathcal{D}_s$ of the OCV routine, where data in $\mathcal{D}_s$ have been normalized with standard (1) and object-wise normalization, employing the GTB algorithm with all the $d = 10$ features[3]. The results show how the proposed ON scheme attains better overall results: this is somewhat expected, given the different features distribution between different items.

3.4. Feature selection

Algorithm 2 and Algorithm 3, described in Section 2.3, are employed to select a reduced subset of features. Algorithms 2 computes the ranking of the features using the same GTB implementation and hyperparameters values as reported in Section 3.3. Instead, Table 3 reports the results of the first five iterations (in addition to the case where all $d = 10$ features are considered) of the Algorithm 3, with performance threshold $\varepsilon = 95\%$.

Algorithm 3 selects $r_{\hat{\omega}}^{(a)}$ as the most important feature, i.e. the RMS of the estimated EMA item speed during the acceleration movement phase. With only this feature, we attain an OCV classification accuracy of 93.9%.

[3]The GTB implementation is the `LightGBM` package (Ke et al., 2017) with a softmax cost function. Code: `lgb.LGBMClassifier(objective = 'multiclass', n_estimators = 500, learning_rate = 0.1, max_bin=100, max_depth = 5, min_data_in_leaf = 75)`

Table 2: Comparison of object-wise cross validation classification accuracy with standard and ON normalization.

Test set	OCV accuracy with standard normalization [%]	OCV accuracy with ON [%]
m_1	85	97
m_2	77.5	96.5
m_3	66	94.25
m_4	93.5	100
m_5	90.5	96.75
m_6	83.75	97.5
Average	**82.7**	**97**

Table 3: Average OCV classification accuracy using GTB varying the number of features considered.

No. features	Selected features	OCV accuracy [%]
1	$r^a_{\hat{\omega}}$	93.9
2	$r^a_{\hat{\omega}} + r^a_i$	95.7
3	$r^a_{\hat{\omega}} + r^a_i + \min^{(c)}\{i(t)\}$	96.7
4	$r^a_{\hat{\omega}} + r^a_i + \min^{(c)}\{i(t)\} + \max^{(c)}\{\hat{\omega}(t)\}$	96.8
5	$r^a_{\hat{\omega}} + r^a_i + \min^{(c)}\{i(t)\} + \max^{(c)}\{\hat{\omega}(t)\} + r^c_i$	96.8
10	**All d features**	**97**

The second most important feature is $r_i^{(a)}$, i.e. the RMS of the EMA item current during the acceleration movement phase. With these first two features, the OCV accuracy becomes $95.7\% > \varepsilon$. Thus, we chose to retain a number $q = 2$ over a total of $d = 10$ features. Furthermore, selecting only features from *the same motion phase* (i.e. acceleration phase) allows to reduce the number of data that need to be processed and stored for features computation.

Table 3 results suggest that the acceleration phase is the most important one for predicting the weight class. This has also a physical interpretation: since an higher weight corresponds to an higher force that opposes to the direction of motion, this translates into a variation of the speed and current in the acceleration phase.

Thus, the reduced raw and normalized train datasets $\mathcal{Z}, \tilde{\mathcal{Z}}$, are composed by the $\mathbf{z}_j \in \mathbb{R}^{2\times 1}$ and $\tilde{\mathbf{z}}_j \in \mathbb{R}^{2\times 1}$ observations respectively, with $j = 1, \ldots, N$. The same rationale holds for the test datasets $\mathcal{Z}_T$ and $\tilde{\mathcal{Z}}_T$.

Remark 4. *The two most important features in Table 3 have a clear physical interpretation. Moreover, their selection is in accordance with the physical interpretation of the phenomenon: in fact, in our setting, it is during the acceleration phase that the dynamics of the actuator may be observed. The load inertia is known to influence the actuator dynamics, and so the gate mass may be estimated from those features.*

3.5. Best algorithm selection

We evaluated $A = 25$ classification algorithms from the MatLab `Classification Learner toolbox` and the GTB model of Section 3.3. Figure 11 and Figure 12 report, respectively, the OCV classification performance and prediction time for each algorithm. The performance is about 95% for almost all algorithms, but the average prediction times differ considerably. *Linear Discriminant Analysis* (LDA), see (Hastie et al., 2009, Chapter 4), is the best algorithm according to the rationale of Section 2.4[4]. The LDA model assumes that $P(\mathbf{x}|y)$ can be modeled as a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}_g$ and $\boldsymbol{\Sigma}$ are the mean vector and covariance matrix of the g-

[4]The LDA model uses a classification threshold of 0.5 to discerning between on class and another.

th class respectively, with $g = 1, \ldots, G$. Notice that $\mathbf{\Sigma}$ is assumed to be equal for all G classes.

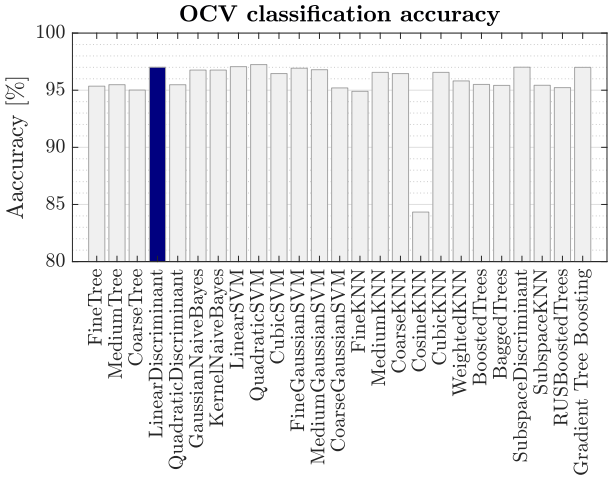

Figure 11: OCV classification accuracy comparison.

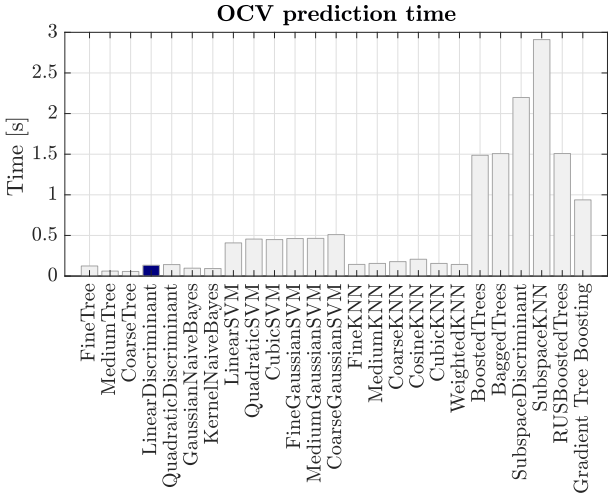

Figure 12: OCV prediction time comparison.

Remark 5. *In this work, only black-box machine learning algorithms are considered, since the aim is to automatically discover the best features, while also evaluating their selection and model assessment procedures. Soft-computing tools like fuzzy systems or graphical models are not employed, due to their complexity in defining their knowledge base, which would require prior information or an higher amount of data. Furthermore, those algorithms are usually more computationally intensive than the considered ones.*

Remark 6. *The proposed Object-wise Cross-Validation (OCV) routine is used many times for different purposes:*

- *the first time in Section 3.3, when performing a comparison between standard and proposed Object-wise Normalization (ON);*
- *a second time in Section 3.4 when performing features selection using Algorithm 2 and Algorithm 3;*
- *a third time in Section 3.5, when the evaluation of different classification algorithms, using the reduced set of features, is performed.*

4. Evaluation of the selected mass classification algorithm for sliding gates

This section presents the experimental evaluations of the chosen LDA classification algorithm, focusing on: (i) classification results on external ambient data from environment e_2; (ii) the effectiveness of the EMA item characterization procedure of Section 2.5 for EOL tests.

4.1. Classification under a different environment

Figure 9 in Section 2.1 showed how a different environment causes a variation in the features distribution. Here, we show the performance of the LDA classifier (trained on data from environment e_1) on the test dataset $\mathcal{Z}_T$ acquired on item m_6 in environment e_2, see Section 3.1.2.

We now assume two distinct cases:

- **standard normalization case:** normalize the train data $\mathcal{Z}$ with standard normalization (1). Denote the normalized train dataset with $\tilde{\mathcal{Z}}^{\text{std}}$. Then, normalize the test set $\mathcal{Z}_T$ with the mean and standard deviation quantities of $\mathcal{Z}$. Call this dataset $\tilde{\mathcal{Z}}_T^{\text{std}}$.

- **object-wise normalization case:** normalize the train data $\mathcal{Z}$ with ON. Denote the normalized train dataset with $\tilde{\mathcal{Z}}$. Suppose that the characterization procedure of Section 2.5 correctly assigned the data in $\mathcal{Z}_T$ to the cluster of item m_6. Then, normalize the test set $\mathcal{Z}_T$ with the mean and standard deviation quantities of features from m_6. Call this dataset $\tilde{\mathcal{Z}}_T$.

Table 4 and Table 5 report the classification confusion matrices of the test data in $\tilde{\mathcal{Z}}_T^{\text{std}}$ and $\tilde{\mathcal{Z}}_T$, with the LDA classifier trained on *all the N data* available in $\tilde{\mathcal{Z}}$ (i.e. on all experiments in environment e_1), normalized according to each normalization case as described above.

The results show that the proposed ON approach is effective in achieving better classification results, *even on data from a different environment*, if the EMA item characteristics are similar (or, in this case, equal) to one of the EMA items used in the training phase.

Table 4: Confusion matrix on the test dataset $\tilde{\mathcal{Z}}_T^{\text{std}}$ with $N_T = T \cdot W = 80$ experiments on item m_6 in environment e_2, obtained using LDA with standard normalization.

		Predicted Class			
		Light	Medium	Medium Heavy	Heavy
True Class	Light	20			
	Medium	10	10		
	Medium Heavy		19	1	
	Heavy		2	14	4

4.2. Evaluation of the procedure for new EMA item characterization

As shown in Table 2, Table 4 and Table 5, the EMA item characteristics play a considerable role to correctly predict the weight class of its actuated gate. Here, we investigate the effectiveness of the approach

Table 5: Confusion matrix on the test dataset $\tilde{\mathcal{Z}}_T$ with $N_T = T \cdot W = 80$ experiments on item m_6 in environment e_2, using LDA with proposed object-wise normalization.

		Predicted Class			
		Light	Medium	Medium Heavy	Heavy
True Class	Light	19	1		
	Medium		20		
	Medium Heavy			20	
	Heavy				20

proposed in Section 2.5 and Algorithm 5 to characterize a newly produced EMA item m^* with feasible End Of Line (EOL) experiments.

Let $\bar{r}_i^{(a)}$ and $\bar{r}_{\hat{\omega}}^{(a)}$ be the average values of the $r_i^{(a)}$ and $r_{\hat{\omega}}^{(a)}$ features, respectively. Figure 13 depicts the $q = 2$ dimensional space composed by $\bar{r}_i^{(a)}$ and $\bar{r}_{\hat{\omega}}^{(a)}$ as denoted by (3), so that $\bar{\mathbf{z}}_s = [\bar{z}_s^{[1]}, \bar{z}_s^{[2]}]$, where $\bar{z}_s^{[1]} = \bar{r}_{\hat{\omega}}^{(a)}$ and $\bar{z}_s^{[2]} = \bar{r}_i^{(a)}$ are the values computed on the features of the s-th training item in $\mathcal{M}$, in the training environment e_1.

Visual inspection of the data suggests that the EMA items can be grouped into $C = 3$ clusters, with centroids $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3 \in \mathbb{R}^{q \times 1}$.

The validity of Algorithm 5 is assessed by varying the $N_s^* = T^* \cdot W$ number of EOL experiments for a new EMA item m^*, where $T^* < T$ is the number of experiments performed for each of the $W = 4$ weight classes. Then, we evaluated the percentage of correct assignments of each EMA item (now described by its $\bar{\mathbf{z}}_s$ vector computed using only its N_s^* experiments) to its cluster, following Algorithm 4.

Algorithm 5 is applied as follows:

1. fix the number T^* of EOL experiments for each weight class;
2. randomly select an item m^* such that $m^* \in \mathcal{M}$;
3. randomly extract T^* experiments of the item m^*, for each of the W weight classes;

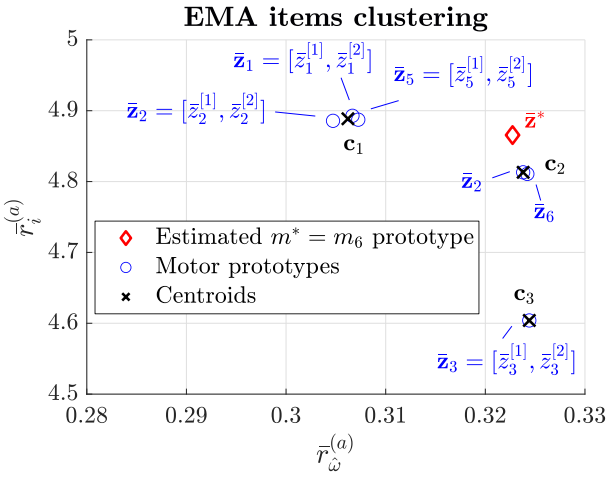

Figure 13: EMA items prototypes used for the characterization of a new item m^*, grouped into $C = 3$ clusters.

4. compute the point $\bar{\mathbf{z}}^* \in \mathbb{R}^{q\times 1}$, consisting in the average of the $q = 2$ selected features over all the $N_s^* = T^* \cdot W$ observations, see (4a);
5. assign $\bar{\mathbf{z}}^*$ to its nearest cluster κ^* with centroid $\mathbf{c}_{\kappa^*} \in \mathbb{R}^{q\times 1}$;
6. check if $\mathbf{c}_{\kappa^*}$ corresponds to the cluster of EMA item m^*, which is known from step 2.

For each choice T^*, the procedure is repeated 2000 times. The main aim of these steps is to define a number of tests T^* that maximizes the clustering accuracy, using only the available dataset without performing another experiment. The evaluation procedures proceed as follows: (i) first, a motor is randomly chosen from the set of available ones; (ii) then, the EOL characterization procedure is simulated by drawing T^* experiments of the selected motor. After that, (iii) clustering is applied and its accuracy evaluated. Thus, if the data gathered from the simulated EOL procedure are not enough, the clustering accuracy will be poor. On the other hand, finding a correct trade-off between the number of experiments and the clustering accuracy is fundamental to keep the time required to perform the EOL tests at an acceptable level. Figure 14 reports the percentage of correct assignments of m^* to its true cluster, as function of the total number of experiments N_s^* on m^*. It can be noticed how selecting a number of $T^* = 3$ experiments for each weight class (so that $N_s^* = 12$) allows to obtain a percentage of correct clustering assignments slightly greater than 80%.

With this choice, we can reduce the number of required EOL experiments on a new EMA item from $N_s = 80$ to $N_s^* = 12$, thus greatly speeding up the characterization procedure of new items with EOL tests. In fact, each experiment takes about 6 seconds, for a total of 72 seconds.

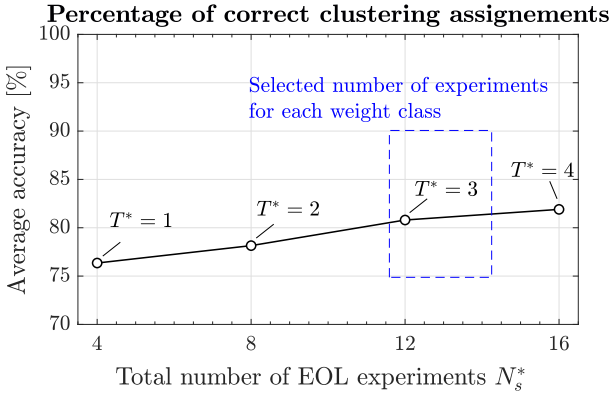

Figure 14: Nearest neighbour clustering assignment performance varying the total number of EOL experiments. Each point represents the average of 2000 simulations.

4.3. A real-world characterization and classification example

We now test our approach as it would be used in a real-world situation, see Figure 15. We retrain from scratch the LDA model using the selected $q = 2$ features on the experiments from EMA items in $\mathcal{M}^- = \{m_1, m_2, m_3, m_4, m_5\}$ in environment e_1. We left the EMA item m_6 in environment e_1 as if it would be the new EOL item m^*, so that $m^* = m_6$. This means that only 5 points are present in Figure 13, and $\mathbf{c}_2 = \bar{\mathbf{z}}_2$. The prototype vector $\bar{\mathbf{z}}^*$ of m^* is computed as in (4a) with a total of $N_s^* = 12$ experiments from those performed on m_6 in environment e_1, since we selected $T^* = 3$. The point $\bar{\mathbf{z}}^*$ is then assigned to a cluster κ^*. This cluster is related to the normalization quantities (6). The prototype of m^* for

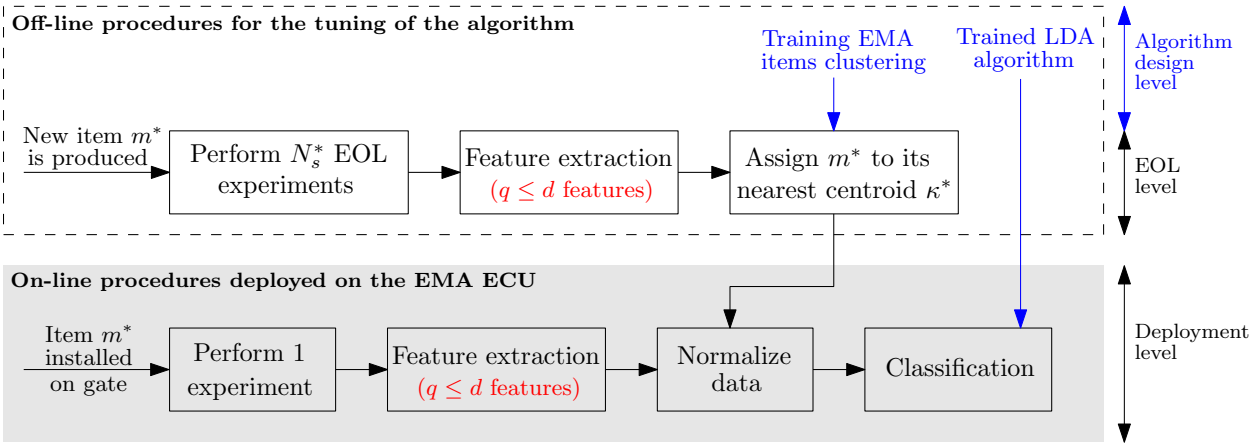

Figure 15: Summary of off-line and online steps for mass classification.

a random sampling of N_s^* experiments is reported in Figure 13 as the red diamond.

Assume now that item m^* gets deployed to a sliding gate, usually in an external environment. We would want to normalize the features computed from its measurements with (6). These data corresponds to the dataset $\mathcal{Z}_T$, measured in environment e_2, where the same EMA item m_6 was employed. Thus, we normalize the data $\mathcal{Z}^* = \mathcal{Z}_T$ with (6), obtaining the normalized test dataset $\tilde{\mathcal{Z}}^*$.

To test the effectiveness of the combined clustering characterization and classification procedures, the LDA algorithm is employed to classify the normalized data $\tilde{\mathcal{Z}}^*$. The whole procedure is repeated 2000 times, randomly varying the subset of T^* experiments extracted from the T available ones from m_6 in environment e_1. Then, a new prototype $\bar{\mathbf{z}}^*$ is computed and assigned to its cluster, so that new normalization quantities are employed to normalize and classify $\mathcal{Z}^*$.

Table 15 shows that, when a correct cluster assignment is made (85% of the cases), the classification accuracy is satisfactory[5]. When the cluster assignment is not correct (15% of the cases), the accuracy drops of about 15%. This drop in performance is however not serious, considering that we are dealing with a 4 classes classification problem.

Table 6: Classification accuracy on test data after EMA item characterization.

	No. of cases	**Classification accuracy**
Correct cluster assignment	1700 ($\approx 85\%$)	99%
Wrong cluster assignment	285 ($\approx 15\%$)	73%

The encouraging results from Table 5 and Table 6 were obtained by performing $T = 20$ experiments for $W = 4$ weights and $M = 6$ motors, for a total of $N = 480$ training data, see Section 3.1.2. However, a different application may require a different number of training data for a sufficiently accurate classification result.

4.4. Analysis of the computational complexity

The computational complexity of the overall procedure is evaluated by considering its different phases and operations. The considered phases are:

[5]We emphasize that, since the classes are perfectly balanced, the accuracy indicator is fine for assessing model performance.

1. classifier design;
2. load classification;
3. train of the EOL system;
4. EOL clustering.

Each applicative phase consists of several computational operations:

A. computation of the features;

B. object-wise normalization;

C. features selection;

D. object-wise cross-validation;

E. features ranking;

F. prediction with LDA;

G. compute sample mean and sample variance;

H. cluster assignment.

Table 7 shows the computational complexity of the method's phases. For each operation, we report the processing time and memory requirement[6]. As an example, the processing time of the feature extraction operation, during the classifier design phase, is 33.68 s, with a memory requirement of 2.304 MB. This is the result of the multiplication of 600 samples (6 s sampled with 100 Hz), 6 motors, 4 weights, 20 movements and the use of 64 bits for representing the floating numbers on the laptop. As expected, the classifier design phase requires the highest memory and processing time.

Table 7: Analysis of the system complexity.

Phase name	Operation name	Processing time [s]	Memory [kB]
Classifier design	**A.**	33.68	2304
	B.	0.82	38.4
	C.	67.59	38.4
	D.	17.73	7.68
Classification	**A.**	0.01	1.6
	B.	$7.6 \cdot 10^{-5}$	0.016
	F.	0.1	0.016
Train of EOL system	**A.**	5.27	38.4
	G.	0.1	7.68
	H.	1.35	7.68
EOL clustering	**A.**	$9.9 \cdot 10^{-2}$	19.2
	G.	$2 \cdot 10^{-3}$	0.192
	H.	0.04	0.192

5. Conclusions

This paper presented a design and evaluation procedure for training a machine learning model under dataset shift for low-cost electro-mechanical actuators. In particular, we devised a fast characterization procedure that can be easily performed by an operator at the end of the production line, in order to obtain a more reliable classification result. We employed a slightly modified version of the data normalization and cross validation methods, tailored to the investigated applicative context. The approach was tested on the mass classification of a sliding gate, using only measurements from the gate actuator. The devised algorithm is a trade-off between performance and computational time, since it should run on the ECU of the actuator. Future research is devoted to a more extensive testing of the methodology.

[6]Referred to a laptop with these characteristics: Intel Xeon E-2176M, 32 GB of RAM and Nvidia Quadro P600.

References

Angeloni, F., Ermidoro, M., Previdi, F., Savaresi, S.M., 2015. A friction estimation approach to fault detection in electromechanical systems. IFAC-PapersOnLine 48, 720–725. doi:10.1016/j.ifacol.2015.09.612. 9th IFAC Symposium on Fault Detection, Supervision andSafety for Technical Processes SAFEPROCESS 2015.

Arlot, S., Celisse, A., et al., 2010. A survey of cross-validation procedures for model selection. Statistics surveys 4, 40–79.

Baur, C., Wee, D., 2015. Manufacturing's next act. Accessed: 2018-01-29.

European Committee for Standardization CEN/TC 33, UNI EN 12453:2017, 2017. Industrial, commercial and garage doors and gates - safety in use of power operated doors - requirements and test methods.

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. Annals of statistics , 1189–1232.

Hastie, T., Tibshirani, R., Friedman, J., 2009. The elements of statistical learning: data mining, inference, and prediction. Springer Science & Business Media.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International conference on machine learning, PMLR. pp. 448–456.

Jayalakshmi, T., Santhakumaran, A., 2011. Statistical normalization and back propagation for classification. International Journal of Computer Theory and Engineering 3, 1793–8201.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y., 2017. Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems 30, 3146–3154.

Lamnabhi-Lagarrigue, F., Annaswamy, A., Engell, S., Isaksson, A., Khargonekar, P., Murray, R.M., Nijmeijer, H., Samad, T., Tilbury, D., Van den Hof, P., 2017. Systems & control for the future of humanity, research agenda: Current and future roles, impact and grand challenges. Annual Reviews in Control 43, 1–64.

Mazzoleni, M., Previdi, F., Scandella, M., Pispola, G., 2019. Experimental development of a health monitoring method for electro-mechanical actuators of flight control primary surfaces in more electric aircrafts. IEEE Access 7, 153618–153634. doi:10.1109/ACCESS.2019.2948781.

Mazzoleni, M., Rito, G.D., Previdi, F., 2021. Electro-Mechanical Actuators for the More Electric Aircraft. Springer International Publishing. doi:10.1007/978-3-030-61799-8.

Moreno-Torres, J.G., Raeder, T., Alaiz-RodrÃguez, R., Chawla, N.V., Herrera, F., 2012. A unifying view on dataset shift in classification. Pattern Recognition 45, 521 – 530. doi:10.1016/j.patcog.2011.06.019.

Pan, F., Converse, T., Ahn, D., Salvetti, F., Donato, G., 2009. Feature selection for ranking using boosted trees, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management, Association for Computing Machinery, New York, NY, USA. pp. 2025–2028. doi:10.1145/1645953.1646292.

Pearl, J., 2009. Causality. Cambridge university press.

Raeder, T., Chawla, N.V., 2009. Model monitor (m 2): Evaluating, comparing, and monitoring models. The Journal of Machine Learning Research 10, 1387–1390.

Strang, G., 2019. Linear algebra and learning from data. Wellesley-Cambridge Press Cambridge.

Subbaswamy, A., Schulam, P., Saria, S., 2019. Preventing failures due to dataset shift: Learning predictive models that transport, in: The 22nd International Conference on Artificial Intelligence and Statistics, PMLR. pp. 3118–3127.

Theodoridis, S., 2020. Machine learning: a Bayesian and optimization perspective, 2nd edition. Academic press. doi:10.1016/C2019-0-03772-7.

Yang, Y., Wu, X., Zhu, X., 2008. Conceptual equivalence for contrast mining in classification learning. Data & Knowledge Engineering 67, 413–429.