Time-series in Hyper-parameter Initialization of Machine Learning Techniques

Tomáš Horváth[1,4][0000−0002−9438−840X],
Rafael G. Mantovani[2][0000−0001−9564−106X], and
André C. P. L. F. de Carvalho[3][0000−0002−4765−6459]

[1] Faculty of Informatics, ELTE - Eötvös Loránd University,
Pázmány Péter sétány 1/C, 1117 Budapest, Hungary
tomas.horvath@inf.elte.hu
[2] Federal Technology University - Paraná, Campus of Apucarana,
Rua Marcílio Dias, 635 - Jardim Paraíso, 86812-460 Apucarana - PR, Brazil
rafaelmantovani@utfpr.edu.br
[3] Institute of Mathematical and Computer Sciences, University of São Paulo,
Avenida Trabalhador São Carlense, 400 - Centro, 13566-590 São Carlos - SP, Brazil
andre@icmc.usp.br
[4] Institute of Computer Science, Faculty of Science, Pavol Jozef Šafárik University,
Jesenná 5, 040 01 Košice, Slovakia
tomas.horvath@upjs.sk

Abstract. Initializing the hyper-parameters (HPs) of machine learning (ML) techniques became an important step in the area of automated ML (AutoML). The main premise in HP initialization is that a HP setting that performs well for a certain dataset(s) will also be suitable for a similar dataset. Thus, evaluation of similarities of datasets based on their characteristics, named meta-features (MFs), is one of the basic tasks in meta-learning (MtL), a subfield of AutoML. Several types of MFs were developed from which those based on principal component analysis (PCA) are, despite their good descriptive characteristics and relatively easy computation, utilized only marginally. A novel approach to HP initialization combining dynamic time warping (DTW), a well-known similarity measure for time series, with PCA MFs is proposed in this paper which does not need any further settings. Exhaustive experiments, conducted for the use-cases of HP initialization of decision trees and support vector machines show the potential of the proposed approach and encourage further investigation in this direction.

Keywords: Automated ML · Metalearning · PCA · DTW.

1 Introduction

The growing popularity of machine learning (ML) in various application domains and the shortage of data scientists has raised the demand for automated ML (AutoML) [7]. A special focus of AutoML is on configuring the hyper-parameters

(HPs) of ML techniques, a task belonging to the family of black-box optimization problems [11], often approached by heuristics ranging from very simple methods, such that random search, to more advanced ones, such that sequential model-based optimization (SMBO) or biologically inspired techniques, eg. particle swarm optimization (PSO). It is important to note that these heuristics need to perform a certain number of iterations to arrive at recommendation in case of a new dataset. Moreover, the performance of these heuristics depends on the initial selection of the HP settings from which these models start their computations [15]. Another important aspect is that these heuristics possess their own HPs (eg. the population size or the surrogate model) which would also need some fine-tuning to perform in the most optimal way.

Another approach to recommend optimal HP setting for a ML technique on a given dataset is to use meta-learning (MtL) [5]. The main premise of MtL is that knowledge and experience gained from previous applications of various ML techniques on different datasets can be employed to recommend the optimal HP setting in case of a new dataset. In this context, "knowledge" and "experience" are represented by a so-called meta-model that captures the relation between the characteristics of a dataset and the predictive performance of ML techniques w.r.t. various HP settings.

The first step in the use of MtL is the representation of each dataset by a vector of features, often named meta-features (MFs) which describe important aspects of a dataset and are used as input attributes in MtL. The next step is to record the predictive performance of ML techniques w.r.t. their HP settings on these datasets. This performance measure will be the target attribute for MtL. Finally, a meta-model induced by employing classification or regression techniques on the created meta-data can be used to predict the most adequate HP setting for a ML technique on a new dataset. However, not only the ML techniques deriving the meta-model but, also, many MF extraction techniques have their own HP settings which should be fine-tuned as well.

This study proposes a simple, yet efficient MtL approach based on Principal Component Analysis (PCA) MF vectors, denoted PCA-MF here, for real-time recommendation and initialization of HP settings of ML techniques. The proposed approach utilizes dynamic time warping (DTW), a similarity measure well-known in time-series analysis (see [4]), to compute the similarity of two PCA-MFs. The use of DTW is, according to our knowledge, new in MtL. The proposed approach has no own HPs, thus, there is no need to an additional fine-tuning. To empirically evaluate the proposed approach, we simulated the use-case of recommending HP settings for Support Vector Machine (SVM) and Decision Tree Induction (DT) algorithms applied to a new dataset. Experiments using 50 real-world datasets are performed comparing the proposed approach (employing a simple k-Nearest Neighbor (kNN) meta-model) with various baseline approaches.

2 PCA Meta-features

Eight main MF types (categories) are documented in the literature as introduced in Tab. 2. Another type of MFs explores statistical correlations in the dataset's predictive attribute values. For such, PCA can be used and the resulting MFs are called PCA-MF in this paper. Since PCA-MF is in the main focus of this paper, the underlying model will be introduced in a more formal way in this section.

2.1 Principal Component Analysis

According to [13], PCA finds a linear transformation of attributes in a dataset such that the information present in the data is maximally preserved in the sense of minimum squared error. Thus, PCA can be seen as a feature extraction method assigning simple weights to the extracted (latent) features.

To formally show how PCA works, let $\mathbf{D} \in \mathbf{X} \times \mathbf{Y}$ be a pre-processed (i.e. standardized and binarized) dataset such that $\mathbf{X} \in \mathbb{R}^{r \times c}$ represents the predictive attributes and $\mathbf{Y} \in \mathbb{R}^{r \times t}$, the target attribute. For simple regression, binary classification and multi-class classification tasks $t = 1$ while in case of multi-label classification $t > 1$. PCA takes $\mathbf{X}$ as input and returns a matrix $\mathbf{W} \in \mathbb{R}^{c \times c}$, the i-th column $\mathbf{w}_i = (w_{i_1}, w_{i_2}, \ldots, w_{i_c}) \in \mathbb{R}^c$ of which is the i-th transformation vector of weights that maps $\mathbf{X} \in \mathbb{R}^{r \times c}$ to the i-th principal component $\mathbf{p}_i = \mathbf{X}\mathbf{w}_i^T \in \mathbb{R}^r$ $(1 \leq i \leq c)$. Together with $\mathbf{W}$, PCA returns a vector $\mathbf{e} = (e_1, e_2, \ldots, e_c)$ which elements, the eigenvalues, express the importance of the respective principal components. Eigenvalues are related to the standard deviation in $\mathbf{X}$ projected to the corresponding principal components, and represent the proportion of variance in $\mathbf{X}$ explained by the corresponding principal components. The values in $\mathbf{e}$ are in decreasing order, i.e. for each $a, b \in \{1, \ldots, c\}$ such that $a < b$, $e_a \geq e_b$. The proportion of variance π_i in the data, explained by the corresponding ith principal component, is computed as $\pi_i = \frac{e_i}{\sum_{i=1}^{c} e_i}$ and the approach proposed in this paper is based on a MF vector

$$\mathbf{m}^{pca} = (\pi_1, \pi_2, \ldots, \pi_c) \tag{1}$$

2.2 Related Work on PCA-MF

To explain how related approaches work, let us consider that two MF vectors $\mathbf{m}_i^{pca} = (\pi_1, \pi_2, \ldots, \pi_{c_i})$ and $\mathbf{m}_j^{pca} = (\pi_1, \pi_2, \ldots, \pi_{c_j})$ extracted from datasets $\mathbf{D}_i$ and $\mathbf{D}_j$, respectively, may have different lengths, i.e. $c_i \neq c_j$. However, the use of MtL requires[5] a fixed number of predictive attributes, therefore, the characterization of a dataset must result in a fixed number of MFs. The usual solution when the number of MFs vary for different datasets is to aggregate the MFs of the same type into one or more values resulting in one or more MF(s).

[5] Since a meta-model is learned by traditional ML techniques.

The ratio d/c was used in [2] as a MF, where d is the number of principal components that explains 95% of the variance in the dataset[6]. The skewness and the kurtosis of the first principal component $\mathbf{p}_1$ are added to the previous PCA MF d/c in [8]. The use of the minimum and the maximum eigenvalue (e_1 and e_c) and the proportion of variance π_1 explained by the first principal component were proposed as MFs in [9].

The price of aggregation to only one, two or three values, as described above, is a possible loss of useful meta-information about the dataset. The use of 22 PCA-based MFs was proposed in [1], such that the 10 histogram bin values of the proportion of variance explained by each principal component[7], together with their normalized values, the proportion of variance explained by the first eigenvalue (π_1 from the Eq. 1) and the ratio d/c from [2].

A more general definition of the *histogram MF vector* introduced in [1], by allowing an arbitrary number b of bins for the histograms, can be defined as

$$\mathbf{m}^{his} = (\theta_1, \theta_2, \ldots, \theta_b) \tag{2}$$

such that $\theta_i = \sum_{j=1}^{c} \delta\left(\frac{i-1}{b} < \pi_j \leq \frac{i}{b}\right)$ for $1 \leq i \leq b$ and $\delta(x) = 1$ if the expression x is true, otherwise $\delta(x) = 0$.

We propose another method, slightly different variation of the histogram MF vector, called the *cut-point MF vector* in this paper which, first, computes the vector $\mathbf{m}^c = (\vartheta_1, \vartheta_2, \ldots, \vartheta_c)$ of cumulative proportion of variances from the vector $\mathbf{m}^{pca}$ where $\vartheta_i = \sum_{j=1}^{i} \pi_j$ for $1 \leq i \leq c$. Next, $\mathbf{m}^c$ is aggregated into the vector

$$\mathbf{m}^{cup} = (\kappa_1, \kappa_2, \ldots, \kappa_b) \tag{3}$$

where $\kappa_i = \arg\min_j \ \vartheta_j \geq \frac{i}{b}$, for $1 \leq i \leq b$. In other words, the indices of those eigenvalues are returned which correspond to the cumulative proportion of variance that first reach the given cut points $\frac{1}{b}, \frac{2}{b}, \ldots, \frac{b}{b}$ of the interval $[0, 1]$. Since κ_b is, in general, equal to the number of eigenvalues (principal components), the introduced cut-point aggregation preserves the meta-information about the number of predictive attributes in the dataset. As a result, a MF vector $\mathbf{m}^{pca}$ of length c is aggregated into a MF vector $\mathbf{m}^{his}$ of length b such that b is the same for all datasets.

As far as we know, the histogram MF vector method has not become widely used in MtL. Also, we could not find any approach similar to the cut-point MF method in the literature.

[6] According to the vector $\mathbf{m}^c = (\vartheta_1, \vartheta_2, \ldots, \vartheta_c)$, $d = min\{i \,|1 \leq i \leq c,\ \vartheta_i \geq 0.95\})$, and c is the number of attributes in the dataset.

[7] Basically, a 10-bin histogram of the values of $\mathbf{m}^{pca}$ from the Eq. 1

Algorithm 1 kNN based recommendation of initial hyper-parameters

1: **procedure** RECOMMENDHP($mfe, \mathbf{D}, \mathcal{H}^\star, \mathcal{M}, k, sim$)
2: $\mathbf{m} \leftarrow$ getMetaFeatures($mfe, \mathbf{D}$)
3: $\{i_1, i_2, \ldots, i_k\} \leftarrow$ getNearestNeighbors($\mathbf{m}, \mathcal{M}, k, sim$)
4: $\mathbf{h} \leftarrow @(\mathbf{h}^\star_{i_1}, \mathbf{h}^\star_{i_2}, \ldots, \mathbf{h}^\star_{i_k}),\ \{\mathbf{h}^\star_{i_1}, \mathbf{h}^\star_{i_2}, \ldots, \mathbf{h}^\star_{i_k}\} \subseteq \mathcal{H}^\star$ ▷ aggregation @
5: **return h**

3 The Proposed Approach

Let $\mathbf{h} = (h_1, \ldots, h_k) \in \mathcal{H}$ be a particular HP setting for a ML algorithm (often called as base-learner) where $\mathcal{H} = \mathcal{H}_1 \times \cdots \times \mathcal{H}_k$ is the admissible domain of possible HP settings. Let the function $f : \mathcal{H} \times \mathcal{D} \rightarrow \mathbb{R}$ represent the accuracy of the base learner with the HP setting $\mathbf{h} \in \mathcal{H}$ on the dataset $\mathbf{D} \in \mathcal{D}$, where $\mathcal{D}$ is the set of datasets. Let $\mathcal{D}^\star = \{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_n\} \subset \mathcal{D}$ be the set of those datasets, called train datasets for MtL, for which the "best" HP settings $\mathbf{h}^\star_i = \underset{\mathbf{h} \in \mathcal{H}}{arg\,max}\, f(\mathbf{h}, \mathbf{D}_i)$ with relation to the base-learner are already known, computed off-line by an arbitrary HP tuning technique and stored in $\mathcal{H}^\star = \{\mathbf{h}^\star_1, \mathbf{h}^\star_2, \ldots, \mathbf{h}^\star_n\}$. In this off-line phase, the MF vectors related to each $\mathbf{D}_i \in \mathcal{D}^\star$ of the form $\mathbf{m}^{pca}$, $\mathbf{m}^{his}$ or $\mathbf{m}^{cup}$ (Eqs. 1, 2 or 3, respectively), denoted here as $\mathbf{m}_i = (m_{i_1}, m_{i_2}, \ldots, m_{i_q})$, are extracted and stored in $\mathcal{M} = \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_n\}$ where q is the number of MFs.

The k-Nearest Neighbor (kNN) approach is a commonly used algorithm in MtL [5], mainly because it is simple and works fast when the amount of data, number of datasets and MFs is not large, which is the usual case in MtL.

The generic kNN based recommendation process of initial HP values consists of three major steps, as illustrated in the Algorithm 1: First, the MF vector $\mathbf{m} = (m_1, m_2, \ldots, m_q)$ is extracted from a "new" dataset $\mathbf{D} \in \mathcal{D}$ according to a given MF extraction approach mfe[8] corresponding to a certain MF type. Second, the indexes $i_1, i_2, \ldots, i_k$ of the k-nearest neighbors $\mathbf{D}_{i_1}, \mathbf{D}_{i_2}, \ldots, \mathbf{D}_{i_k}$ of $\mathbf{D}$, using a MF similarity function sim[9], are found. Finally, the best HP settings $\mathbf{h}^\star_{i_1}, \mathbf{h}^\star_{i_2}, \ldots, \mathbf{h}^\star_{i_k}$ recorded for the k-nearest neighbors of $\mathbf{D}$ are aggregated (line 4) using some aggregation function[10] @ to get the returned results.

3.1 Utilizing Dynamic Time Warping

As pointed out before, aggregating MFs brings a risk of losing possibly useful information. However, the use of the standard MtL approaches, such as the kNN with vector similarity measures[9] or any other ML meta-learner (eg. Random

[8] We are using all the eight MF types described above as well as their different combinations in our experiments as baselines.

[9] In case of MF vectors of the same size (eg. $\mathbf{m}^{his}$ and $\mathbf{m}^{cup}$) we were experimenting with well-known vector similarity measures such as Euclidean distance, inner product, cosine similarity and Pearson correlation.

[10] This study uses a simple average as an aggregation function, however, any other aggregation function, like a weighted average, can be used, too.

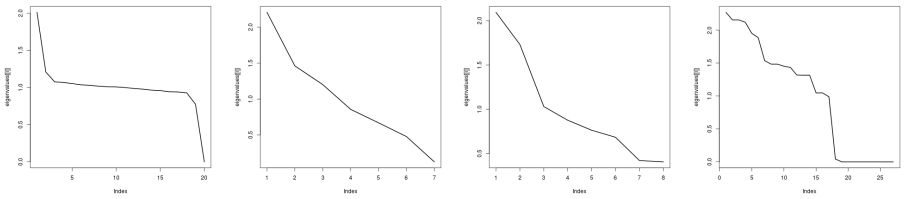

Fig. 1. An example of PCA-MFs $\mathbf{m}^{pca}$ for four datasets. The graphs show the eigenvalues (y-axis) for the principal components (x-axis) which numbers correspond to (different) numbers of attributes in these datasets.

Forest), would work only with MF vectors of equal lengths. Moreover, MF aggregation would require to specify (ie. fine-tune) the length of the MF vectors. Finally, the choice of the optimal MF types to extract and use is data dependent.

Thus, we propose a novel approach, utilizing Dynamic Time Warping (DTW) [12], to measure the similarity between two MF vectors $\mathbf{m}_i^{pca}$ and $\mathbf{m}_j^{pca}$, defined in the Eq. 1. DTW is an edit-distance like algorithm for measuring the similarity of two time series by finding the best matching alignment between them even if the two time series on its input have different lengths. This is the main motivation of our proposal to employ DTW for measuring the similarity between two datasets represented by $\mathbf{m}_i^{pca}$ and $\mathbf{m}_j^{pca}$ of different lengths resulting from different number of attributes in the corresponding datasets $\mathbf{D}_i$ and $\mathbf{D}_j$.

The two MF vectors $\mathbf{m}_i^{pca}$ and $\mathbf{m}_j^{pca}$ are, basically, two decreasing sequences expressing the (speed of) down-grade of the proportion of variance along the latent directions (ordered w.r.t. their importance) computed via PCA in the corresponding two datasets $\mathbf{D}_i$ and $\mathbf{D}_j$. Since each sequence is composed by the principal components extracted from a dataset and the principal components usually preserve the main characteristics of a dataset, similar sequences are expected to be mapped to the same meta-target label.

An example situation with four datasets is illustrated in Fig. 1 where, according to the "shapes" of the PCA-MFs, the middle two datasets are the most similar to each other. Thus, it is assumed that the optimal HP settings (i.e. meta-target labels) would be also similar for these two datasets.

Summarizing, in our proposal, the kNN meta-learner (Alg. 1) is applied to the PCA-MF vectors $\mathbf{m}^{pca}$ (Eq. 1) with the *sim* function being the DTW.

The similarity of two datasets is a subjective matter and can be seen, expressed and measured in various ways what can be seen also in the colorful palette of various MF types (see Tab. 2). PCA-MFs represent another view on dataset similarity looking at this matter in different context. It is important that nor DTW nor PCA, in its basic form, has no HPs to set up beforehand. The only HP to fine-tune would be the k in the kNN model, however, this would have to be set up using other types of MFs as well.

Table 1. (Multi-class) Classification datasets used in the experiments and their main characteristics: number of rows (r), columns (c) and classes (t).

No.	Name	r	c	t	No.	Name	r	c	t
1	**acute-infl.-nephr.**	99	6	2	2	**analcatdata_lawsuit**	263	4	2
3	**appendicitis**	106	7	2	4	**autoUniv-au6-cd1-400**	400	40	8
5	**banknote-authentication**	1348	4	2	6	**breast-cancer-wisconsin**	463	9	2
7	**breast-tissue-4class**	105	9	4	8	**bupa**	341	6	2
9	**climate-simul.-craches**	540	20	2	10	**cloud**	108	6	4
11	**connect.-mines-vs-rocks**	208	60	2	12	**dermatology**	366	34	6
13	**ecoli**	336	7	8	14	**fertility-diagnosis**	100	9	2
15	**glass**	213	9	6	16	**habermans-survival**	289	3	2
17	**hayes-roth**	93	4	3	18	**heart-dis.-proc.-hun.**	293	13	2
19	**hepatitis**	155	19	2	20	**horse-colic-surgical**	300	27	2
21	**indian-liver-patient**	570	10	2	22	**ionosphere**	350	33	2
23	**iris**	147	4	3	24	**leaf**	340	15	30
25	**led7digit**	146	7	10	26	**leukemia-haslinger**	100	50	2
27	**mammographic-mass**	689	5	2	28	**monks3**	438	6	2
29	**movement-libras**	330	90	15	30	**parkinsons**	195	22	2
31	**pima-ind.-diab.**	768	8	2	32	**planning-relax**	176	12	2
33	**prnn_crabs**	200	7	2	34	**qualitative-bankr.**	103	6	2
35	**robot-failure-lp4**	116	90	3	36	**saheart**	462	9	2
37	**seeds**	210	7	3	38	**spect-heart**	228	22	2
39	**statlog-heart**	270	13	2	40	**teaching-assist.-eval.**	110	5	3
41	**thoracic-surgery**	470	16	2	42	**thyroid-newthyroid**	215	5	3
43	**tic-tac-toe**	958	9	2	44	**user-knowledge**	403	5	5
45	**volcanoes-e3**	1276	3	5	46	**voting**	281	16	2
47	**wdbc**	569	30	2	48	**wholesale-channel**	440	7	2
49	**wine**	178	13	3	50	**wpbc**	198	33	23

4 Experiments

The experiments use two base-learners which differ in their learning paradigms as well as the number and types of HPs. These base-learners are i) Support Vector Machine (SVM) with the RBF kernel with two HPs (both real valued) to initialise, and, ii) Decision Tree (DT) with nine HPs (real-valued and Boolean) to initialise.

Since the motivation of this study is a MtL approach suitable for on-line initialization of HP values, and class imbalance can occur, the experiments measure the predictive performance regarding balanced accuracy [6] and runtime. A total of 50 multi-class classification datasets, from the UCI Machine Learning Repository[11] are used in the experiments as introduced in Tab. 1.

Various approaches were used as baselines for the experiments, differing in the simplicity and strategy adopted to initialize or tune HPs. The first batch of used baselines contains four HP tuning techniques:

- The default HP setting for SVM and DT provided by `R` packages `e1071` and `RWeka`, respectively, denoted as "DF". Although this is the simplest baseline, it can present a good performance in some situations [10].
- The best HP setting found by an exhaustive Random Search in the HP space, denoted by "RS", suggested in [3] as a good alternative for HP tuning. The number of trials in RS was set to 2500 for SVM and 5000 for DT.
- The best HP setting found by Particle Swarm Optimization [16], denoted as "PSO". For SVM, the maximum number of evaluations was set to 2500. For DT, the maximum number of evaluations was 5000. The default HP

[11] http://archive.ics.uci.edu/ml/

Table 2. MF types, and their abbreviations (Abbr), identified in the literature. # denotes the number of different MFs belonging to a given MF type/category.

MF type	Abbr	#	Description
Simple	SL	17	Simple measures
Statistical	ST	7	Statistics measures
Inf. theoretic	IT	8	Information theory measures
Landmarking	LM	9	Performance of some ML algorithms
Model-based	MB	17	Features extracted from decision trees
Time	TI	5	Execution time of some ML algorithms
Complexity	CO	14	Measures analyzing complexity
Complex Network	CN	9	Complex network property measures

settings recommended by the corresponding `R` libraries for SVM and DT, respectively, were added to the initial populations.
- The best HP setting found by Sequential Model-based Optimization [14], denoted here as "SMBO". For both SVM and DT, the maximum number of evaluations (budget) was set to 200 and Random Forest (RF) was used as surrogate learning model.

These baselines are for HP tuning and not HP initialisation since they are computationally too expensive for on-line scenarios. However, since they perform exhaustive (random or sophisticated) search, they can be good references to measure the performance of HP initialisation approaches, such that, we can see how close the initialised HPs are to the tuned ones. The other family of baselines involves the following, traditional MtL approaches for HP initialisation:

- HP settings recommended by a kNN based meta-learner applying traditional vector similarity measures[9] to MF vectors $\mathbf{m}^{cmf} = (m_1, m_2, \ldots, m_l)$, representing various MF type combinations, having the same length l for all the 50 datasets. Here, all the $2^8 - 1 = 255$ different combinations of the 8 MF types presented in the Tab. 2 are considered corresponding to 255 different MF vectors[12]. The lengths l of these MF vectors vary from 5 (only the TI MFs are used) to 86 (all the MFs are used). These baselines are denoted as "NN-CMF", an acronym for kNN based meta-learner with a certain Combination of MF types.
- HP settings recommended by a kNN based meta-learner applying traditional vector similarity measures[9] to MF vectors $\mathbf{m}^{his}$ and $\mathbf{m}^{cup}$, defined in Eqs. 2 and 3, respectively, having the same length b for all the 50 datasets. These baselines are denoted as "NN-HIS" and "NN-CUP", respectively.

The experiments compare the performance of the proposed approaches with the baselines. They also investigate the influence of different HP settings and extensions of the compared approaches on their predictive accuracy. For such,

[12] Where either all or none of the MFs belonging to a certain MF type were present in a MF vector, according to the given combination of MF types.

Algorithm 2 One complete cycle of experiments

1: HP tuning using SMBO, PSO, RS, DF ▷ Computing $\mathcal{H}^\star$
2: extraction of MF ($\mathbf{m}^{pca}$,$\mathbf{m}^{his}$, $\mathbf{m}^{cup}$, $\mathbf{m}^{cmf}$) ▷ Computing $\mathcal{M}$
3: recommendation of HP settings for the base-learners ▷ Computing $\mathbf{h}$
4: computation of the accuracy of the base-learners with the recommended HP settings averaged on the folds of 5-fold cross-validation of a given dataset.

an experimental procedure performing a complete cycle of recommendation is adopted for the experiments. The steps of this cycle are listed in the Alg. 2 (the comments in the lines refer to the given parts and notations in Alg. 1).

This cycle is computed for all datasets in a leave-one-out manner, i.e. for each dataset the cycle is performed such that the other 49 datasets are used as "train" datasets for MtL and HP tuning. The balanced accuracy averaged over the 5 folds will be denoted as "cycle-accuracy" of a dataset w.r.t. some base-learner and HP setting recommender. For each dataset, nested cross-validation is utilized for tuning the HPs of a base-level learner (SVM and DT) on this dataset in case of SMBO, PSO and RS with 10 outer and 3 inner folds. The whole cycle presented in the Alg. 2 is run 30 times for each "test" dataset, and the average of the 30 computed[13] "cycle-accuracy" values is returned what will be denoted as "overall-accuracy" values.

The proposed approach, ie. using $\mathbf{m}^{pca}$ MF vectors and the DTW similarity measure, is denoted as "DTW" here.

The datasets and the source codes for the proposed approach as well as for the experiments (eg. Algs. 1 and 2) are publicly available[14] for further use.

4.1 Results

The relative performance of the tested approaches, when compared to each other, is similar for all the four choices for the HP tuning strategy. Fig. 2 and 3 illustrate the averaged "overall-accuracies" across all the different settings of complete cycles (see Alg. 2). The proposed approach (DTW) as well as the aggregated PCA MF approaches (NN-HIS and NN-CUP) performed slightly better than the CMF approaches with regard to the number of clear wins.

In the Fig. 4, the extraction times for different MF extraction approaches related to the used datasets are illustrated. SL MFs are the most faster to extract, however, these are very simple MFs and usually not performing well if only these are used alone without any other MF types. The following most fastest to extract MFs are the proposed PCA-MFs.

[13] This was done because of the stochastic nature of the used HP tuning algorithms (SMBO, PSO and RS), thus, to get more accurate statistics about the performance of the used approaches.

[14] https://github.com/rgmantovani/TimeSeriesHPInitialization

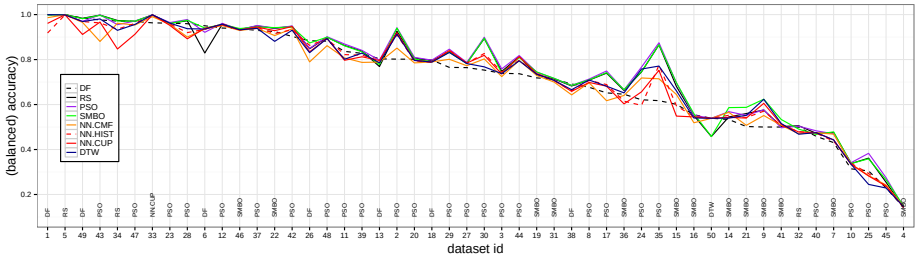

Fig. 2. Performance of different approaches for SVM base-learner: Averaged "overall-accuracies" across all the different settings of complete cycles (see Alg. 2). Clear winner approaches are marked in the bottom line (abbreviations are placed in the vertical).

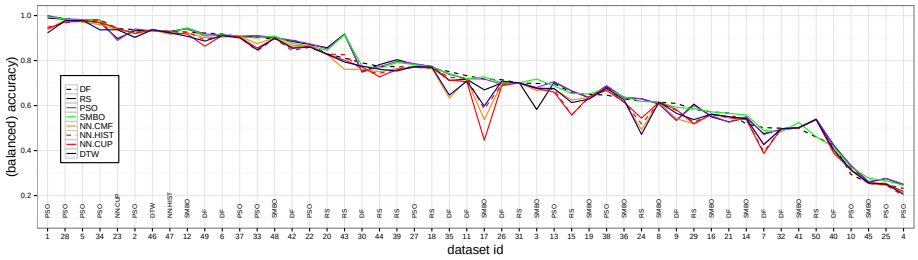

Fig. 3. Performance of different approaches for DT base-learner: Averaged "overall-accuracies" across all the different settings of complete cycles (see Alg. 2). Clear winner approaches are marked in the bottom line (abbreviations are placed in the vertical).

4.2 Discussion

There are two important issues, mentioned in the introduction, regarding the traditional MtL, approaches: First, as showed in the experiments, the choice of a suitable combination of MF types for HP recommendation seems to be data dependent. Second, the extraction of some MF types usually has a high computational cost. It is important to remember that in the experiments, either all the MF of a certain MF type were considered or were excluded completely from the consideration. No MF selection procedure was performed on the set of all the 86 MFs listed in Tab. 2 what would probably result in better recommendation, but with additional computational cost. The most time-consuming process is to find the suitable combination of MFs which has the larger range of values, i.e. 2^n where n is the number of MFs to be considered (86 in this case). The use of MF selection methods can significantly increase the performance, however, is time-consuming and not appropriate for on-line scenarios where the (initial) HP settings need to be delivered in real time.

Regarding the number of bins in case of NN-CUP and NN-HIS approaches, the b parameter, experiments indicate that a tuning is required, since, in general, there is no clear better choice. However, since the reasonable values to use are

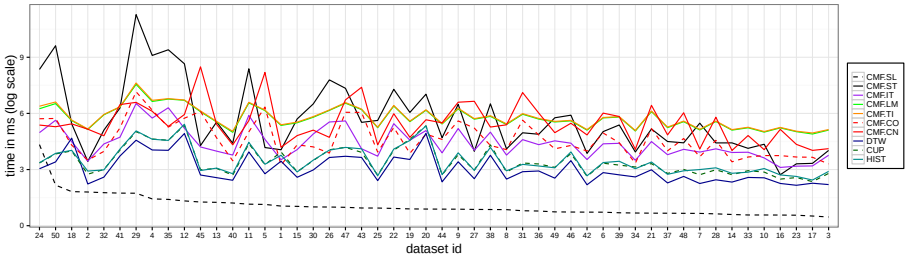

Fig. 4. MF Extraction times, on a logarithmic scale, for the used datasets and MF extraction approaches.

the dividers of 100, the tuning has to deal with a small set of values (i.e. 1, 2, 4, 5, 10, 20, 25, 50 and 100) compared to the before mentioned range of 2^n for the possible combinations of MF types. It is also worth to mention that PCA needs to be performed only once for a dataset to obtain various MF vectors for different values of b.

Finally, DTW does not need to tune any own HPs what, considering it's competitive performance, makes it a good choice for scenarios where HP settings need to be initialized fast.

5 Conclusions

This paper proposed and investigated a fast approach to HP Initialisation of Ml algorithms utilizing PCA-MFs and DTW. To the best of authors' knowledge, the proposed DTW approach is novel.

The proposed approach was evaluated and compared to various baselines in the use-case scenario of initialisation of HP for SVM and DT using kNN meta-learner. Experiments were performed on 50 real-world datasets.

The results showed that the proposed approach presents a good predictive accuracy when compared with various baseline approaches with a run time faster than the used baselines. Also, the performance of the base-learners using the initialised HPs are very close to the performance of the base-learners using optimized HPs via exhaustive HP tuning approaches.

The proposed DTW approach has no parameters to tune and is sufficiently accurate and fast to be used in on-line scenarios where HP values need to be recommended in real time.

Acknowledgment "Application Domain Specific Highly Reliable IT Solutions" project has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the Thematic Excellence Programme TKP2020-NKA-06 (National Challenges Subprogramme) funding scheme.

References

1. Amasyali, M.F., Ersoy, O.K.: A study of meta learning for regression. Tech. rep., ECE Technical Reports. Paper 386., Purdue e-Pubs, Purdue University (2009)
2. Bardenet, R., Brendel, M., Kégl, B., Sebag, M.: Collaborative hyperparameter tuning. In: Proceedings of the 30th International Conference on Machine Learning. vol. 28, pp. 199–207. JMLR Workshop and Conference Proceedings (2013)
3. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. J. Mach. Learn. Res. **13**, 281–305 (2012)
4. Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: AAAI Workshop on Knowledge Discovery in Databases. pp. 359–370 (1994)
5. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: Metalearning: Applications to Data Mining. Springer-Verlag Berlin Heidelberg, 1 edn. (2009)
6. Brodersen, K.H., Ong, C.S., Stephan, K.E., Buhmann, J.M.: The balanced accuracy and its posterior distribution. In: Proceedings of the 2010 20th International Conference on Pattern Recognition. pp. 3121–3124. IEEE Computer Society (2010)
7. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Advances in Neural Information Processing Systems 28, pp. 2944–2952. Curran Associates, Inc. (2015)
8. Feurer, M., Springenberg, J.T., Hutter, F.: Using meta-learning to initialize bayesian optimization of hyperparameters. In: International Workshop on Meta-learning and Algorithm Selection co-located with 21st European Conference on Artificial Intelligence. pp. 3–10 (2014)
9. Janssens, J.H.: Outlier Selection and One-Class Classification. Ph.D. thesis, Tilburg University, Netherlands (2013), TiCC PhD Dissertation Series No.27.
10. Mantovani, R.G., Rossi, A.L.D., Vanschoren, J., Bischl, B., de Carvalho, A.C.P.L.F.: To tune or not to tune: Recommending when to adjust svm hyper-parameters via meta-learning. In: 2015 International Joint Conference on Neural Networks. pp. 1–8 (2015)
11. Mantovani, R.G., Horváth, T., Cerri, R., Junior, S.B., Vanschoren, J., de Leon Ferreira de Carvalho, A.C.P.: An empirical study on hyperparameter tuning of decision trees (2019)
12. Ratanamahatana, A., Keogh, E.: Everything you know about dynamic time warping is wrong. In: 3rd Workshop on Mining Temporal and Sequential Data, in conjunction with 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (2004)
13. Sharma, A., Paliwal, K.K.: Fast principal component analysis using fixed-point algorithm. Pattern Recognition Letters **28**(10), 1151–1155 (2007)
14. Snoek, J., Larochelle, H., Adams, R.P.: Practical bayesian optimization of machine learning algorithms. In: Pereira, F., Burges, C., Bottou, L., Weinberger, K. (eds.) Advances in Neural Information Processing Systems 25, pp. 2951–2959. Curran Associates, Inc. (2012)
15. Wistuba, M., Schilling, N., Schmidt-Thieme, L.: Hyperparameter search space pruning – a new component for sequential model-based hyperparameter optimization. In: European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 104–119. Springer International Publishing (2015)
16. Yang, X.S., Cui, Z., Xiao, R., Gandomi, A.H., Karamanoglu, M.: Swarm Intelligence and Bio-Inspired Computation: Theory and Applications. Elsevier Science Publishers B. V., 1st edn. (2013)