ISSN 1870-4069

Towards a Transfer Learning Strategy in Full Model Selection Algorithm for Temporal Data Mining

Nancy Pérez Castro[1], Héctor Gabriel Acosta Mesa[2]

[1]University of Papaloapan, Engineering and Technology, México

[2]University of Veracruz, Artificial Intelligence Research Center, México

nperez@unpa.edu.mx, heacosta@uv.mx

Abstract. Traditional machine learning techniques were designed for training for scratch depend on the current feature-space distribution. In many real applications, the fact to obtain new data for training and rebuilds models could become expensive or impossible. Therefore, from a lifelong machine learning conceptualization, transfer learning can be indeed beneficial to speed up the time it takes to develop and train a model by reusing an isolated pre-training setting as a starting point for another target domain, especially when multiple tasks and hyper-parameter optimization are considered, such as a full model selection approach. This document presents an early transfer learning strategy based on a decision tree powered by full models for temporal databases trained in an isolated way with different search methods. The proposed transfer learning strategy is capable to suggesting the starting point and the search method adopted by the full model selection approach.

Keywords: transfer learning, full model selection, temporal databases.

1 Introduction

Humans have the innate capacity to transfer knowledge across tasks. In this regard, the acquired experience can be utilized in the same way to solve related tasks. Therefore, the more connected tasks, the easier it is to transfer or cross-utilize the knowledge. Concerning Computer Science, particularly, Data Mining (DM) and Machine Learning (ML) fields, have been inspired by how human beings learn and transfer knowledge to simulate those behaviors through algorithms.

However, although significant progress in knowledge engineering in both DM and ML algorithms has achieved, most of them have been traditionally designed to work in isolation. The isolated training means that the built models are focused on specific tasks and depend on the current feature-space distribution.

Therefore, if the distribution changes, the models need to be rebuilt from scratch using newly collected training data.

In many real applications, the fact to obtain new data for training and rebuilds models could become expensive or impossible. In order to overcome these issues, researchers and scientists turn the gaze toward knowledge transfer or transfer learning, whose purpose is centered on the need for lifelong machine-learning methods that retain and reuse previously learned knowledge. According to Goodfellow et al. [5], transfer learning is described as *the situation where what has been learned in one setting is exploited to improve generalization in another environment.*

Research on transfer learning has attracted more attention in the last decades in different ways such as life-long learning, multi-task learning, knowledge transfer, inductive transfer, knowledge consolidation, context-sensitive learning, knowledge-based inductive bias, meta-learning, incremental/cumulative learning, and recently Auto-machine learning (AutoML) or Full Model Selection.

The specialized literature on learning transfer highlights three important research issues:

– What to transfer?
– How to transfer?
– When to transfer?

Regarding these issues, Pan and Yang [8], suggest a classification of transfer learning approaches according to three sub-settings: a) *Inductive transfer learning*, the target task is different from the source task, while the source and target domains can be the same or not. b) *Transductive transfer learning*, the source and target tasks are the same, while the source and target domains could be the same. c) *Unsupervised transfer learning*, the target task is different but related to the source task focus on unsupervised learning tasks in the target domain.

In related literature, it has been observed that in transfer learning approaches, especially inductive transfer learning, it is possible to transfer instances, feature representation, parameters, and relational-knowledge. Regarding related works of transfer learning, most of the research has been developed within the framework of artificial neural networks where multi-task can be involved [10].

In the context of the Full Model Selection (FMS) problem, where multiple task and hyper-parameters optimization are involved, the transfer learning has not been explored. Since one of the benefits of transfer learning is that it can speed up the time it takes to develop and train a model by reusing these settings as a starting point for another scene. The transfer learning strategy turns out to be an attractive option to accelerate the search for complete models.

In this regard, an early proposal of transfer learning in the frame of an FMS algorithm for temporal data mining tasks is presented. The remaining sections of this document are organized as follows. In Section 2, a brief theoretical background of FMS in temporal data is given. In Section 3, the employed methodology is described. Then, in Section 4, the preliminary results are outlined. Finally, Section 5 presents conclusions and future work.

2 Background

2.1 Full Model Selection Problem

Full Model Selection problem (FMS) refers to all aspects of automating the machine learning process, including model selection and hyper-parameter optimization for carrying on different tasks in an incremental way. In order to produce a suitable combination of methods which help to classify or predict a new data within a fixed computational cost, FMS can involve two remarkable concerns: (1) no single method performs well on each dataset, and (2) some methods work appropriately base on its hyper-parameter optimization. Both issues are known as Algorithm Selection and Model Selection problems [3, 11].

This work tackles the FMS problem in temporal databases, mainly in time-series, as a single-objective optimization problem through an evolutionary wrapper approach, where population-based metaheuristics or single point-search metaheuristics can be used [7, 2, 9]. An instance of FMS problem for temporal data consists of finding a suitable combination of smoothing, time-series representation, instances reduction, and classification methods with the setting of their related hyper-parameters. The FMS problem is expressed in Equation 1, where a set of algorithms $\mathcal{A} = A^1, ..., A^n$ with their related hyper-parameters $\theta = \{\theta_1, ..., \theta_m\}$ and labeled training data $\mathcal{D} = \{(x_1, y_1), ..., (x_n, y_n)\}$ are used to find the optimal generalized performance, for which, the training data is split up into disjoint training $\mathcal{D}^i_{train}$ and validation $\mathcal{D}^i_{valid}$ datasets which are evaluated by loss function $\mathcal{L}$ in an isolated training through k-cross-validation method:

$$\mathcal{A}^*, \boldsymbol{\theta}_* \in \underset{A^J \in A, \theta \in A^j}{\arg\min} \sum_{i=1}^{K} \mathcal{L}(\mathcal{A}^j_\theta, D^i_{train}, D^i_{valid}). \tag{1}$$

One of the advantages of solving the FMS problem by evolutionary wrapper is the capacity of manipulate multiple task and the hyper-parameter optimization at the same time. However, the main disadvantages of this approach are the high computational cost during the isolate training and the absence of reusing trained models for other data domains. Therefore, to treat those drawbacks, and inspired by lifelong machine learning (LML) paradigm [12], a set of experiments to find a strategy of transfer learning within the framework of the FMS algorithm is carried out.

3 Methodology

In this section, the adopted general methodology is described and presented in Figure 1. The considered instance space $\mathcal{X}$ is a set of time-series databases, taken of a well-known benchmark [6]. The considered FMS approach is widely described in [9], and general behavior is presented in Algorithm 1. This approach can be works under two different metaheuristics structures, based-population or a single point optimizer [1, 13].

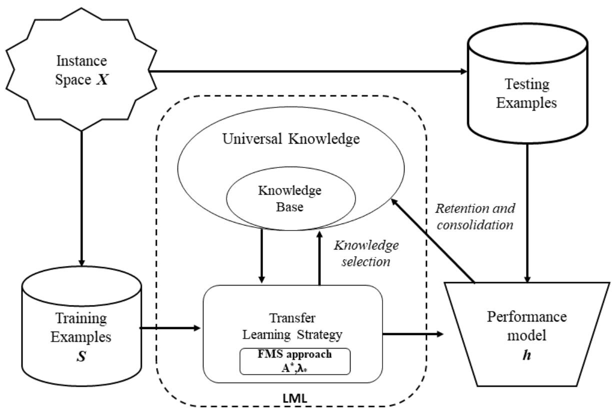

Fig. 1. Graphical representation of interaction between specific instance space, FMS approach and transfer learning strategy under LML paradigm.

The based-population structure is guided by μ-Differential Evolution algorithm (four variants are evaluated) while the single point search operates under local search (two different solution encoding are considered). Candidate solutions are composed of a combination of methods for smoothing, time-series representation, instance selection and classification with associated hyper-parameters. The original FMS approach was designed to train in an isolated way where all candidate solutions are evaluated according to the cross-validated miss classification rate, depending on the available database.

At the end of the evolutionary process, the best solution is obtained for each database. In order to build a strategy to transfer learning under LML paradigm, a knowledge base is needed, that a long term will be interacting with universal knowledge. In this work, the knowledge base is powered by a decision tree building from the best full models obtained during the isolated training of the FMS algorithm in its different versions per each database. Concerning the LML framework, the FMS algorithm must use the decision tree to determine the starting point for the training stage of a new database, as well as the recommended metaheuristic. Therefore, with the transfer learning strategy based on a decision tree, the cost to generate full models for new instances of temporal databases is expected to be lower than training from scratch. So far, the retention and consolidation are not considered in this early proposal.

4 Experiment and Results

This section presents a set of experiments realized to build and evaluate the early transfer learning strategy for FMS algorithm. The experimentation is presented in two stages: (1) A comparison of the final statistical results of variants of FMS

Algorithm 1 General behavior of FMS algorithm

```
Require: A (Pool of available methods), $\theta$ (Set of involved hyper-parameters), $M$ (Metaheuristic
    as optimizer), $fitness$ (Fitness Function), $D_{train}$ (Train dataset), $D_{valid}$ (Validation dataset),
    $D_{test}$ (Test dataset),
1: Set $maxItera$ % Maximum number of iterations
2: Set $i = 0$
3: Set $M$ % which can be population-based or single point optimizer
4: Randomly generate initial solution(s) % if $M$ is population-based a set of solutions are generated,
   other side one initial solution is generated
5: while $i < maxItera$ do
6:     Starts optimization process through population-based or single point search
7:     A fitness function is used for evaluating
8:     Special operators are involved (crossover, mutation, selection or neighborhood generator)
9: end while
10: Get best final solution $\vec{s}$ % involves a suitable combination of methods and their hyper-
    parameters
11: Evaluate $\vec{s}$ on $D_{test}$
```

algorithm to build a based decision tree knowledge base and (2) preliminary results of the transfer learning strategy.

4.1 Stage 1: Knowledge Base Building

Six versions of FMS algorithm were trained in an isolated environment where eight temporal databases (Table 1) were used. The four firsts versions of FMS algorithm correspond to population-based option while the two rests are compatible with the single-point search option. In all cases, the termination condition was 3000 evaluations, and five independent runs were carried out. The configuration used by each involved metaheuristics is described as following, based on [2, 14]:

P-DEMS versions: $iter = 500$, $NP = 6$, $CR = 0.1$, $F = 0.9$, $N = 2$ and $R = 10$.
S-LSMS versions: $iter = 500$ and $N_k = 6$.

Table 3 shows the final numerical results obtained by the six FMS version training in a isolated way. The population-based versions were known as P-DEMS1 to P-DEMS4 and the single point search versions were named as S-LSMS1 and S-LSM2. The reported values correspond to the average of five trials evaluated in the testing set of each database. A non-parametric Friedman test was used [4] for multiple comparison among FMS versions. Friedman test converts numerical values to ranks. Thus, it ranks the FMS versions for each problem separately, the best performing algorithm version should have rank 1, the second-best rank 2, etc. When ties are presented, like this case, average ranks are computed.

According to the average ranks, it is observed that SLSMS1 and PDEMS1 were the two best versions, followed by P-DEMS2, S-LSMS2, P-DEMS4 and P-DEMS3, respectively. From this, information a simple log-archive was created. The log-archive contains information related to each database's information (time-series length, number of classes, domain), the full best model obtained by isolated training of the six versions of FMS algorithm for each database, the

Table 1. Time-series databases description.

No.	Name	No. of classes	Training set size	Testing set size	Time-series length	Domain
1.	Beef	5	30	30	470	Spectro
2.	CBF	3	30	900	128	Simulated
3.	Coffee	2	28	28	286	Spectro
4.	ECG200	2	100	100	96	ECG
5.	FaceFour	4	24	88	350	Image
6.	Gun_Point	2	50	150	150	Motion
7.	Lightning-2	2	60	61	637	Sensor
8.	Lightning-7	7	70	73	319	Sensor
9.	OliveOil	4	30	30	570	Spectro
10.	Trace	4	100	100	275	Sensor

Table 2. Description of knowledge base attributes.

Attribute	**Description**	Type
length	Time-series length.	Numeric
classes	Number of classes.	Numeric
smooth	Type of selected smoothing method.	Nominal
representation	Type of selected time-series representation method.	Nominal
insReduc	Type of selected instance selection method.	Nominal
error	Misclassification rate of the tested model.	Numeric
Meta	Version of FMS algorithm.	Nominal

average runtime during isolated training for each database and FMS algorithm, the test misclassification rate of each full model and the name of FMS algorithm.

From this, information a simple log-archive was created. The log-archive contains information related to each database's information (time-series length, number of classes, domain), the full best model obtained by isolated training of the six versions of FMS algorithm for each database, the average runtime during isolated training for each database and FMS algorithm, the test misclassification rate of each full model and the name of FMS algorithm. A total of 300 models were stored that gave rise to form a supervised knowledge database, where the name of the FMS algorithm was considered as the class attribute.

Because only two versions of FMS algorithm reported competitive results, the knowledge database was limited to only store the models of these versions. Then, the knowledge base was composed of seven attributes, detailed in Table 2, with 100 different models. A decision tree of Weka was selected to generate a practical and visual way that supports the rules generation that can be incorporated as part of the learning transfer strategy. The accuracy of the decision tree was of 83.10%, and it is presented in Figure 2.

4.2 Stage 2: Adoption and Testing of the Learning Strategy

According to the taxonomy of learning transfer approaches, the proposed strategy, in this work, is classified as *Transductive transfer learning*, because of the

Table 3. Comparison of averaging performance among the six metaheuristics for each database. Values to the right of ± represent the standard deviation and the values in parentheses represent the ranks computed by the Friedman test. Values in **boldface** mean the lowest values found or the best ranking.

Database	P-DEMS1	P-DEMS2	P-DEMS3	P-DEMS4	S-LSMS1	S-LSMS2
Beef	0.053±0.102 (3)	0.087±0.038 (4)	**0.000±0.000 (1.5)**	0.160±0.060 (5)	**0.000±0.000 (1.5)**	0.367±0.227 (6)
CBF	**0.000±0.000 (3)**	**0.000±0.000 (3)**	**0.000±0.000 (3)**	**0.000±0.000 (3)**	**0.000±0.000 (3)**	0.030±0.027 (6)
Coffee	**0.000±0.000 (3)**	**0.000±0.000 (3)**	**0.000±0.000 (3)**	**0.000±0.000 (3)**	0.268±0.157 (6)	0.000+0.000 (3)
ECG200	**0.000±0.000 (2)**	0.800±0.447 (4)	1.000±0.000 (5.5)	1.000±0.000 (5.5)	**0.000±0.000 (2)**	**0.000±0.000 (2)**
FaceFour	**0.000±0.000 (3.5)**	**0.000±0.000 (3.5)**	**0.000±0.000 (3.5)**	**0.000±0.000 (3.5)**	**0.000±0.000 (3.5)**	**0.000±0.000 (3.5)**
Gun_Point	**0.000±0.000 (1)**	0.395±0.221 (4)	0.493±0.000 (5.5)	0.493±0.000 (5.5)	0.388±0.217 (3)	0.212±0.253 (2)
Lightning-2	**0.000±0.000 (3)**	**0.000±0.000 (3)**	**0.000±0.000 (3)**	**0.000±0.000 (3)**	**0.000±0.000 (3)**	0.069±0.154 (6)
Lightning-7	0.766±0.035 (5)	0.762±0.028 (4)	0.786±0.019 (6)	0.761±0.012 (3)	**0.019±0.019 (1)**	0.082±0.046 (2)
OliveOil	0.013±0.030 (2.5)	0.033±0.047 (5)	0.027±0.043 (4)	0.013±0.018 (2.5)	**0.000±0.000 (1)**	0.133±0.122 (6)
Trace	0.800±0.447 (3.5)	0.800±0.447 (3.5)	1.000±0.000 (5.5)	1.000±0.000 (5.5)	**0.000±0.000 (1.5)**	**0.000±0.000 (1.5)**
Average rank	2.950	3.700	4.050	3.950	**2.550**	3.800

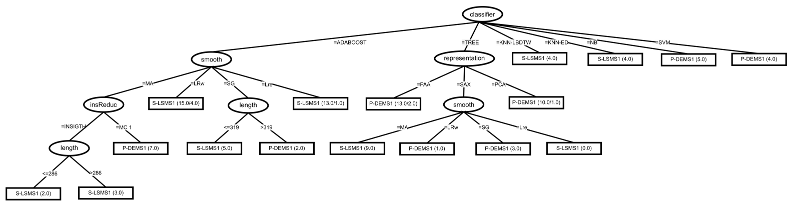

Fig. 2. Decision tree of knowledge base.

source and target tasks are the same, while the source and target domains could be the same or not. Regarding the three principal questions on $what$(Q1), how(Q2)and $when$(Q3) to transfer, these will be described below:

Q1: Setting of full models that includes selected methods and their hyper-parameters optimized. Besides the suggested search engine for continue the training process.

Q2: The pre-trained models that will be the starting point for another dataset will be randomly selected within the knowledge base, as long as the pre-trained models have been used in smaller time series or of the same length as the new database. A set of six different models can be selected as randomly, which will be evaluated by the decision tree. The tree will suggest a class tag for each instance, which corresponds to the type of search engine. Considering the majority vote, If the population-based option (P-DEMS1) is suggested, all models are transferred. Otherside, if the single point optimizer (S-LSMS1) is suggested, only one of the six can be assigned.

Q3: At the beginning of the training process of a new data set that not exists in the knowledge base.

The proposed transfer strategy for FMS algorithm was tested on four databases, the preliminary results are shown in Table 4. Similar behavior was obtained by LML-FMS in two of the four databases against isolated training. The suggested search strategy for theses cases was S-LMS1. Otherside, the significantly worse cases were produced by the P-DEMS1 search engine. An improvement speed up on training was observed when P-DEMS1 was suggested as a search engine.

Table 4. Comparison between the isolated training and proposed approach by transfer learning. IT means isolated training, while LML-FMS means lifelong machine learning for the full model selection. Values in **boldface** mean the significant lowest values.

No.	Name	Classes	TS-Length	.6 Domain	.6 IT	.6 LML-FMS
1.	ECGFiveDays	2	136	ECG	**0.0000**	0.0011
2.	SonyAIBORobotSurface	2	70	SENSOR	0.0032	0.0080
3.	SonyAIBORobotSurfaceII	2	65	SENSOR	**0.0031**	0.0183
4.	ItalyPowerDemand	2	24	SENSOR	0.0264	0.0255

5 Conclusions and Future Work

In this paper, a transfer learning strategy for the FMS algorithm for temporal data mining was presented. The initial knowledge base was built from isolated pre-trained full models, and the transfer learning is based on a decision tree powered by that base. Although isolated training provides better solutions in two of the databases, preliminary results through transfer learning show competitive results, encouraging to extend experiments in other database domains. Therefore, as future work, data complexity measures, test data distribution, or model complexity could be considered into the knowledge base. Besides, to explore other ways to transfer the learning between different temporal domains data.

References

1. Caraffini, F., Neri, F., Poikolainen, I.: Micro-differential evolution with extra moves along the axes. In: Differential Evolution (SDE), 2013 IEEE Symposium on. pp. 46–53 (April 2013)
2. Escalante, H.J., Montes, M., Sucar, L.E.: Particle swarm model selection. Journal of Machine Learning Research 10(Feb), 405–440 (2009)
3. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28, pp. 2962–2970. Curran Associates, Inc. (2015)
4. García, S., Fernández, A., Luengo, J., Herrera, F.: Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Sciences 180(10), 2044 – 2064 (2010), special Issue on Intelligent Distributed Information Systems
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016), http://www.deeplearningbook.org
6. Keogh, E., Zhu, Q., Hu, B., Hao, Y., Xi, X., Wei, L., Ratanamahatana, C.A.: The UCR Time Series Classification/Clustering Homepage (2011), www.cs.ucr.edu/ẽamonn/time_series_data/
7. Momma, M., Bennett, K.P.: A pattern search method for model selection of support vector regression. In: Proceedings of the 2002 SIAM International Conference on Data Mining. pp. 261–274. SIAM (2002)
8. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. on Knowl. and Data Eng. 22(10), 1345–1359 (Oct 2010)

9. Pérez-Castro, N., Acosta-Mesa, H.G., Mezura-Montes, E., Cruz-Ramírez, N.: Towards the full model selection in temporal databases by using micro-differential evolution. an empirical study. In: 2015 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC). pp. 1–6 (Nov 2015)
10. Reddy, T.K., Arora, V., Kumar, S., Behera, L., Wang, Y., Lin, C.: Electroencephalogram based reaction time prediction with differential phase synchrony representations using co-operative multi-task deep neural networks. IEEE Transactions on Emerging Topics in Computational Intelligence 3(5), 369–379 (Oct 2019)
11. Rice, J.R.: The algorithm selection problem. In: Rubinoff, M., Yovits, M.C. (eds.) Advances in computers, vol. 15, pp. 65 – 118. Elsevier (1976)
12. Silver, D.L., Yang, Q., Li, L.: Lifelong machine learning systems: Beyond learning algorithms. In: AAAI Spring Symposium: Lifelong Machine Learning. AAAI Technical Report, vol. SS-13-05. AAAI (2013)
13. Talbi, E.: Metaheuristics: From Design to Implementation. Wiley Series on Parallel and Distributed Computing, Wiley (2009)
14. Viveros-Jiménez, F., Mezura-Montes, E., et al.: Empirical analysis of a micro-evolutionary algorithm for numerical optimization. International Journal of Physical Sciences 7(8), 1235–1258 (2012)